Efficient Call Management in Broadband Networks

Andrew M. Inggs

THESIS PRESENTED IN PARTIAL FULFILMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
AT THE UNIVERSITY OF STELLENBOSCH

Prof. A. E. Krzesinski
December 1999

Declaration

I, the undersigned, hereby declare that the work contained in this thesis is my own original work and that I have not previously in its entirety or in part submitted it at any university for a degree.

Date: 20/10/1999

Opsomming

Die skakeling- en oordragvermoë van kommunikasienetwerke het toegeneem. Terselfdertyd het 'n behoefte ontstaan om kommunikasiedienste te integreer. Dit het gelei tot die ontwikkeling van 'n standaard vir die oordrag van alle telekommunikasiedienste oor 'n algemene netwerk, bekend as ATM ("asynchronous transfer mode").

Dit is belangrik dat die winstempo in breëband ATM-netwerke wat multidiensverkeer dra, optimaal moet wees. Daar bestaan dan ook verskeie tegnieke om die ontwerp van oproeptoegangskontroles en oproeproetebepalingstrategieë te optimeer. Verskeie optimeringstegnieke is in hierdie navorsing ondersoek, en 'n nuwe tegniek, XFG, is ontwikkel.

XFG is 'n klein, effektiewe algoritme vir die bepaling van 'n virtuele-pad konneksie netwerk (VPKN) wat die winstempo optimeer. Die optimale VPKN bied roete skeiding (elke oorsprong-eindpunt paar is verbind deur 'n afsonderlike VPK) en diens integrasie (alle dienste deel 'n VPK).

Hierdie tesis verduidelik die teoretiese basis van XFG. Daarna word die algoritme op verskeie toetsnetwerke toegepas. Die resultate wat sodoende verkry word, word dan gebruik om XFG te evalueer en met ander optimeringstegnieke te vergelyk.

Uit die resultate blyk dat XFG – vir dié netwerke wat ondersoek is – 'n toepaslike algoritme is vir die dinamiese herkonfigurasie van 'n groot VPKN in reaksie op stadige tydskaal variasies in die gelewerde multidiensverkeer. Alternatiewe roetebepaling, of die gebruik van oproepwagtoue, kan saam

met dinamiese herkonfigurasie gebruik word. Sodoende kan die lukraak verskille wat op 'n kort tydskaal tussen aankomende verkeer en oordragvermoë ontstaan, hanteer word. Die kombinasie van dinamiese herkonfigurasie en alternatiewe roetebepaling/gebruik van wagtoue kan benut word om die vereiste diensgehalte vir 'n wye reeks van multidiensverkeer te verkry.

Abstract

Huge increases in switching and transmission capacity coupled with a need to integrate communication services has lead to the development of the asynchronous transfer mode (ATM), a standard for the transport of all telecommunication services over a common network.

This thesis describes optimization techniques used to design call admission controls and call routing strategies which optimize the rate of earning revenue in broadband ATM networks carrying multirate traffics. Several optimization techniques are investigated and a new technique called XFG is presented. XFG is a small efficient algorithm for calculating a virtual path connection network (VPCN) that optimizes the rate of earning revenue. The optimal VPCN provides route separation (each origin-destination pair is connected by a dedicated VPC) and service integration (all service classes share a VPC). The thesis presents the theoretical basis of XFG, followed by applications of the algorithm to several test networks. These results are used to evaluate XFG and to compare XFG to other optimization techniques.

We show that, for the network models under investigation, XFG is a suitable algorithm to dynamically reconfigure a large VPCN in response to slow time-scale variations in the offered multirate traffics. We further show that dynamic reconfiguration can be augmented by alternative routing or call queueing to deal with the short time-scale random mismatches between offered traffic and capacity. The combination of dynamic reconfiguration and alternative routing/call queueing can be used to achieve the required grade of service for a wide range of multirate traffics.

To my brother David

Contents

List of Tables

List of Figures

Acknowledgements

I would like to thank the following people:

- Mom and Dad, for their support and encouragement.
- Corné, for stealing my heart and making the second year of my Masters the best year of my life (so far).
- Tony Krzesinski, for his strong academic example, his encouraging faith in me, and for the many opportunities he gave me.
- Reg Dodds, whose constant enthusiasm for Computer Science was a source of inspiration.
- Pieter de Villiers, who taught me to ask, "Is there a simpler way to do this based on more fundamental principles?"
- Abriëtte Senekal, for keeping the Department running smoothly and being a friend.
- The rest of the staff and students at the University of Stellenbosch who shared time with me, especially those in the Department of Computer Science; each of you taught me something.

I would also like to express my gratitude to the University of Stellenbosch, the trustees of the Harry Crossley Scholarship Fund, Telkom South Africa, and the Foundation for Research Development (now part of the National Research Foundation) for their financial support during the course of my studies.

Chapter 1

Introduction

This chapter describes the context in which this thesis will discuss call management in broadband networks, followed by a brief outline of the chapters to follow.

1.1 Call Management in Broadband Networks

The last decade has seen dramatic increases in switching and transmission capacity, accompanied by an increasing need to integrate communication services – including voice, data, video and multimedia – over the same telecommunication network. These trends have led to the development of the asynchronous transfer mode (ATM), a standard for the transport of all telecommunication services over a common network. ATM is a strong contender for providing integrated services to network users.

ATM is connection oriented – paths (routes) are set up between communicating entities and calls are offered to these routes for connection. The calls are connected if the network has sufficient resources to connect the call without adversely affecting the quality of service offered to the already connected calls. Bandwidth constitutes a major resource in ATM networks. The focus of much recent work has been the concept of *effective bandwidth* [4, 8, 9, 12, 13, 18, 22, 31].

Ross [30] defines effective bandwidth in terms of a linear acceptable admission policy. Assume a link has transmission capacity C and virtual channels (VCs) can be established on this link, where each VC is a member of one of K service classes, service class k having a peak bandwidth requirement of b_k. The *VC profile* is $(n_1, \ldots, n_K)$ where n_k is the current number of established service-k VCs.

A VC profile $\mathbf{n} = (n_1, \ldots, n_K)$ is said to be *allowable* if the quality of service requirements are met for all K services when $\mathbf{n}$ is permanent (that is, no new VC establishments or VC departures). An *allowable admission policy* is one in which an arriving service-k VC is accepted if, and only if, the new VC profile $\mathbf{n}$ would be allowable. An admission policy is said to be a *linear policy* if there exist positive numbers $b_1^e, \ldots, b_K^e$ such that a service-k VC is accepted if and only if

$$\mathbf{b}^e \cdot \mathbf{n} + b_k^e \leq C \tag{1.1}$$

where $\mathbf{b}^e = (b_1^e, \ldots, b_K^e)$. Under a linear allowable admission policy with associated vector $\mathbf{b}^e$ we say that b_k^e is the *effective bandwidth* of service k. Figure 1.1, taken from Ross, shows the admission regions for three allowable policies for a link with two service classes.

When the effective bandwidth approximation holds good, a *loss network* (see Ross [30] and the references therein) provides an appropriate model to study call performance issues in a telecommunication network. A loss network is a collection of resources shared by calls. When a call arrives to find insufficient resources available, the call and its potential revenue are lost. Telecommunication engineers have employed loss networks to model the performance of telephone systems. However, traditional loss models for telephone networks exclude heterogeneity in the calls' bandwidth requirements, the central attribute of ATM networks. Loss networks have therefore been extended to describe multiservice circuit-switched networks and Ross [30] discusses a variety of models for ATM networks based on loss networks and the concept of effective bandwidth.

We will use loss networks to model call admission controls in multiservice

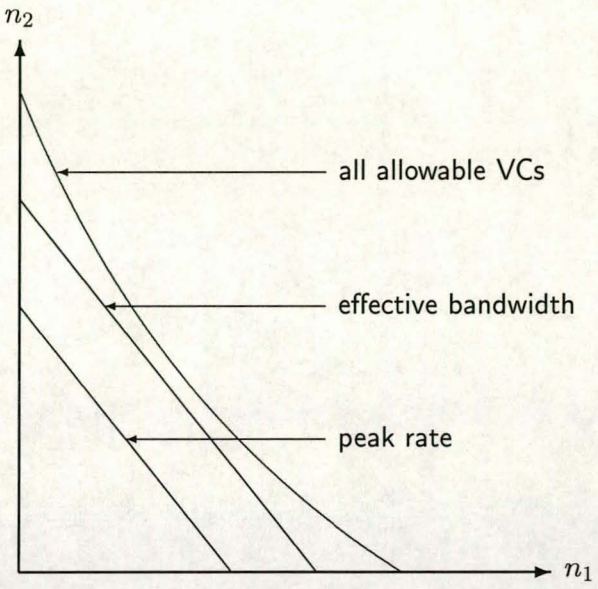

Figure 1.1: The boundaries of the admission regions for three allowable policies

broadband networks. Traffic of a number of different types requiring different bandwidth allocations is offered to each origin–destination pair. The network manager must implement a call admission control scheme to maximize the revenue earned from the network while maintaining agreed quality of service constraints.

Various possibilities exist for such a scheme. At one extreme, there is the complete sharing scheme in which the network manager admits calls of any type provided that there is enough capacity on a route (that is, a set of physical links) connecting its origin and destination. At the other extreme, there is the complete partitioning scheme in which the manager partitions the bandwidth on the physical links to form a logically fully-connected network consisting of virtual path connections. In such a network, traffic of each type traveling between each origin–destination pair has its own allocated capacity. Connections are accepted only if there is sufficient capacity on one of the corresponding virtual path connections. Intermediate schemes are the service partitioning scheme in which physical capacity is partitioned according to traffic type but not according to origin and destination, and the

route partitioning scheme where capacity is partitioned according to origin and destination but not according to traffic type. In the former case, the physical network is partitioned into separate virtual subnetworks, one for each traffic class. In the latter case the network consists of one or more multirate virtual path connections per origin–destination pair.

All networks are subject to traffic variation. If this were not the case, good network dimensioning would be enough to find a configuration that provides near optimal revenue for the operator. Because this is not the case, network operators are left with the task of allocating sufficient resources to cater for the varying traffic demands.

It is important to distinguish between levels at which traffic variations take place. Table 1.1 taken from Arvidsson [1] shows one way of defining the different levels, together with the time scales of each.

The *link level* refers to the variations in the number of potential users, that is, the number of users who are able to use their equipment. This is influenced by office hours, weekends, major sporting events and the like. It can be managed by reallocating capacity to areas of high demand at the appropriate times.

The *session level* refers to variations in the number of active users, that is, the number of users using their equipment. These variations can be regarded as random and therefore described by stochastic means. They are dealt with by statistical multiplexing and routing strategies.

The *burst level* refers to variations in user behaviour, that is, the number of active users currently generating information. Examples include the alter-

Level	Source of variation	Time scale	Management technique
link	user behaviour	hours	capacity reallocation
session	equipment usage	minutes	overflow, routing
burst	information generation	seconds	larger buffers, flow control
cell	cell formation	milliseconds	smaller buffers, policing

Table 1.1: Hierarchical classification of traffic variations.

nation between listening and talking in voice applications, screen updates versus keyboard input in data applications, and the alternation between differentially encoded and full frames in video applications. Variations at this level display both deterministic patterns and random behaviour. Management is typically by statistical multiplexing via buffers on which flow control is applied. The size of the buffer may depend on application demands such as loss, delay, and jitter; and the flow control, for example window schemes and explicit congestion notification, is carried out in software.

The *cell level* refers to variations in user delivery, that is, the number of users currently presenting information to the network. In ATM networks, variations at this level reflect cell adaptation routines carried out by the ATM adaptation layer (AAL). Management proposals include buffers combined with numerous versions of policing, spacing and fast reservation mechanisms. The high transmission rates and traffic integrity demanded of ATM suggest that control must be carried out in hardware and that only limited buffering is meaningful.

Call management comprises call admission control, routing strategies and the techniques used for reallocating capacity in the network. As such, it is concerned with variations at the link and session levels. Burst and cell variations are managed separately, their influence at the call level is restricted to that which can be encapsulated in the effective bandwidth.

Under a given call admission control and routing strategy, we define a capacity reallocation technique as an optimization technique where the network resources are configured to provide an optimal (or near-optimal) rate of earning revenue for the network operator.

1.2 The Problem Addressed in this Thesis

In this thesis we are concerned with network management at the call level. Specifically we are interested in optimization techniques – together with their corresponding call admission controls and routing strategies – that op-

timize the rate of earning revenue. We consider techniques that optimize the network as a whole (centralized) and assume that the network operates under the policy of service integration (statistical multiplexing between traffic classes) where all service classes are carried on a single logical network.

1.3 The Chapters to Follow

Chapter 2 provides an overview of current network optimization techniques which are relevant to the thesis. The technique of Mitra, Morrison and Ramakrishnan [25] is presented in detail as an example of such a technique.

Chapter 3 gives a description of the algorithm XFG, an efficient algorithm for optimizing the virtual path connection network in ATM networks. The algorithm is presented in its mathematical form, without consideration for implementation details.

Chapter 4 provides insights into the computer implementation of XFG. Various features are presented and efficiency considerations discussed. The XFG code can output a trace which details its internal operation. This trace can be processed to display statistics and graphs about the operation of the algorithm, they are given in this chapter.

Finally, chapter 5 presents several experiments performed using XFG. The network performance obtained from dynamic reconfiguration is compared with the results obtained from other call management techniques such as dynamic alternate routing [14] and call queueing.

Chapter 2

A Survey of Network Optimization Techniques

With the emergence of wide area broadband network technologies, much research is being been done into improving network efficiency for the operator while providing an acceptable grade of service for the user. Mathematical models are often used to analyze the performance of communication networks. These models express the performance measures of the network as a function of several variables. The process of network optimization is then one of minimizing or maximizing such a measure within the constraints imposed by the model. In this chapter the virtual path connection network is presented as the context in which optimization is considered. An overview of current optimization techniques is given, followed by the description of one such techniques in detail.

2.1 VPC Networks

Virtual paths (VPs) and virtual path connections (VPCs) are important concepts in ATM networks. A VP recognizes the distinct identity of a traffic stream between two communicating nodes. Cells belonging to a given VP are identified by a common, fixed VP identifier (VPI) and individual connections are identified by a virtual channel identifier (VCI) which is unique to the

VPI. A VPC consists of a series of concatenated VPs and specifies a route to be traversed by a set of virtual channel connections (VCCs) from an originating node through a number of intermediate nodes to a destination node.

Bandwidth is logically assigned to a VPC by reserving a portion of the bandwidth on each transmission link for the exclusive use of the VPC. Capacity can thus be reserved on a VPC in anticipation of call arrivals so that new VCCs can rapidly be set-up along predefined routes: the exclusive ownership of bandwidth implies that the availability of resources can be determined over the entire VPC by executing simple control functions at the end points of the VPC.

A further simplification and speed up can be obtained if service classes are separated into logically distinct VPCs such that each service class has its own logically independent VPC network (VPCN). Alternatively, several service classes can share VPCs and the VPCN can benefit from the resulting multiplexing gain. The bandwidth reserved on a VPC may be expressed in terms of the maximum number of simultaneous connections it can support before violating some grade of service (GOS) constraints such as the cell loss probability. A call admission control (CAC) will accept a request if the number of connections in progress is fewer than this number, and reject it otherwise.

2.1.1 VPCN Management

Typically, the traffic offered to a network will vary over time. These variations may have a regular nature which may be related to successive video frames, common office hours, etc., or exhibit a more random behaviour which may be related to the request and delivery of web pages, public events, etc. A large number of studies in recent years have indicated the existence of variations over many time scales in a whole range of different traffic traces. Both the cause of these variations and the means to handle them depend on the time scale considered. Thus buffers and bandwidth adjustments can be

used to handle fast and slow traffic variations respectively. Fast variations give rise to brief overloads which can be absorbed by buffers; the queued cells will be transmitted when the load drops. Since fast variations are of short duration there is not enough time to adjust the bandwidth. Slow variations, on the other hand, give rise to overloads which persist for enough time to adjust the bandwidth. Buffers are less effective in absorbing lengthy overloads since the buffers will quickly fill up and many cells will be lost.

Three main VPCN management alternatives can be identified: the *fixed* approach, where a VPCN configuration is computed once and for all; the *predetermined* approach, where a set of precomputed VPCN configurations are deployed according to a fixed schedule; and the *dynamic* approach, where VPCN configurations are computed and applied online. The latter strategy can be further subdivided into three different approaches: the *central* approach, where a network management centre collects traffic data, computes VPCN configurations according to global benefit, and makes decisions about whether or not to change an existing configuration; the *local* approach, where each node drops and seizes bandwidth on routes according to demand and availability; and the *distributed* approach, where the nodes negotiate with each other to distribute bandwidth on routes according to mutual benefit.

In the fixed approach, a VPCN is designed according to criteria such as expected demands averaged over some time interval, worst case situations, etc., and semi-permanently configured on the physical network. Fixed VPCNs are simple to manage, but offer little flexibility when it comes to handling erroneous forecasts or significant demand fluctuations. The predetermined approach offers a restricted degree of flexibility by deploying different, precomputed VPCN designs according to given time tables. This allows operators to move resources in the network according to anticipated demand variations, and thus maintain a certain quality of service with fewer resources in total, using a limited number of network management actions which are known in advance. The dynamic approach offers full flexibility by using real time demand estimation and resource control. Online resource allocation enables operators to handle both demand fluctuations and erroneous forecasts, but may result in network management actions which are as frequent

and unpredictable as the traffic conditions they are dictated by.

2.1.2 VPCN Design

The *VPCN design problem* appears in the fixed, predetermined and central dynamic approaches and can be formulated as follows: given a set of nodes which are interconnected by an arbitrary and possibly sparse physical network, and offered traffics for each node pair and service class, find the end-to-end routes and bandwidths per origin–destination pair (and possibly per service class) which maximize network utility. It should be noted that a large variety of definitions for utility can be deployed as the optimization criteria. Several techniques exist for modelling service bandwidth requirements and link capacities. Using the technique of equivalent bandwidth, each service class has a specific bandwidth requirement for each link which is typically a function of the mean and peak bandwidth requirements of the class and the link capacity measured in the same unit of bandwidth. On the other hand the bandwidth of a link can be expressed as the number of cells it can carry per time unit, hence bandwidth partitioning between VPCs means assigning a certain number of cells per time unit to each VPC. Typically, differentiated cell transmission from a switch port is carried out by visiting a set of cell buffers in a cyclic fashion, which means that a fraction of the port bandwidth corresponds to a certain proportion of the visits per cycle. Using this view, the bandwidth may be expressed in integer units.

2.2 The VPCN Design Problem

The following notation will be used in our models. Consider a network with N nodes and L links which connect pairs of nodes. Let $D_{o,d}$ denote the capacity in bandwidth units of the physical link from an origin node o to a destination node d such that $D_{o,d} > 0$ implies the existence of a physical link between nodes o and d. There are S service classes, where service s requires bandwidth $d^s_{o,d}$ on link (o, d). (When the bandwidth requirements are expressed in a link independent fashion, the subscripts o, d are dropped.)

The call arrival stream for service s to O-D pair (o,d) is Poisson, with mean rate $\lambda^s_{o,d}$. The holding period of a call is generally distributed with mean $1/\mu^s_{o,d}$. Let $\rho^s_{o,d} = \lambda^s_{o,d}/\mu^s_{o,d}$ denote the intensity of the offered traffic stream. A call of service s connected between nodes o and d generates revenue at a rate $\theta^s_{o,d}$.

A route $r = (n_1, n_2, \ldots, n_k)$ is a sequence of $k > 2$ different nodes such that $D_{n_i,n_{i+1}} > 0, i = 1, \ldots, k-1$. Let $\mathcal{R}_{o,d}$ denote the set of routes connecting O-D pair (o,d). Note that the direct link from o to d, if it exists, is not referred to as a route. A VPCN is constructed by identifying a set of routes $\mathcal{R} = \bigcup_{(o,d)} \mathcal{R}_{o,d}$ and associating an amount of transmission capacity x_r with each route $r \in \mathcal{R}$. Let $\mathcal{A}_{o,d}$ denote the set of routes that use link (o,d) and let the vector $\mathbf{x} = (x_r)_{r\in\mathcal{R}}$ denote the transmission capacity assignments to all routes in the network. The capacity $C_{o,d}(\mathbf{x})$ of the VPC (o,d) is given by

$$C_{o,d}(\mathbf{x}) = D_{o,d} - \sum_{r\in\mathcal{A}_{o,d}} x_r + \sum_{r\in\mathcal{R}_{o,d}} x_r. \tag{2.1}$$

The first two terms in the above equation refer to the physical link between o and d. If the link exists, it contributes its capacity $D_{o,d}$ to the VPC (o,d) less the capacity allocated to the routes $r \in \mathcal{A}_{o,d}$ which use link (o,d). The third term in equation (2.1) denotes the capacity contributed to the VPC (o,d) which is obtained from the routes $r \in \mathcal{R}_{o,d}$.

Let $B^s_{o,d} = E^s_{o,d}(C_{o,d}(\mathbf{x}))$ denote the blocking probability experienced by a class $s = 1, \ldots, S$ call with effective bandwidth $d^s_{o,d}$ and intensity $\rho^s_{o,d}$ on a VPC of capacity $C_{o,d}(\mathbf{x})$ carrying S services. The total revenue generation rate $F(\mathbf{x})$ for configuration $\mathbf{x}$ is the sum of the revenue generation rates over all O-D pairs and service classes

$$\begin{aligned} F(\mathbf{x}) &= \sum_{o,d} F_{o,d}(C_{o,d}(\mathbf{x})) \\ &= \sum_{o,d,s} \theta^s_{o,d}\,\rho^s_{o,d}(1 - E^s_{o,d}(C_{o,d}(\mathbf{x}))). \end{aligned} \tag{2.2}$$

A configuration $\mathbf{x}$ is *feasible* if all allocations are non-negative and the allocations conform to the limitations of the physical network. Thus for all

routes $r \in \mathcal{R}$

$$x_r \geq 0 \tag{2.3}$$

and for all o, d such that $D_{o,d} > 0$

$$\sum_{r \in \mathcal{A}_{o,d}} x_r \leq D_{o,d}. \tag{2.4}$$

The VPCN design problem is defined as finding the configuration $\mathbf{x}_{\text{opt}}$ that maximizes the revenue earning rate $F(\mathbf{x})$

$$\begin{aligned} F(\mathbf{x}_{\text{opt}}) &= \max_{\mathbf{x}} F(\mathbf{x}) \\ &= \max_{\mathbf{x}} \sum_{o,d,s} \theta^s_{o,d} \rho^s_{o,d} (1 - E^s_{o,d}(C_{o,d}(\mathbf{x}))) \end{aligned}$$

subject to constraints (2.3) and (2.4).

2.3 An Overview of Current Techniques

It is important to distinguish the uncapacitated VPCN design problem where the capacities of the physical links are the decision variables, from the capacitated VPCN design problem where the capacities of the physical links appear as constraints. The uncapacitated problem has been studied by, among others, Røhne et al. [29] who first design a physical network under the assumption that all traffic will be routed on a node-by-node basis, and then configure the VPC highways by cross connecting flows in the physical network according to a cost model. Bauschert [3] also considers the uncapacitated problem and uses a set of iteration loops to simultaneously design a VPCN and a VC routing policy. This scheme relies on an initial pre-selection of paths and includes linear programming models.

A distinction is also made between real valued configurations and integer valued configurations. In the former, bandwidth is regarded as a resource which can be shared in any fashion while in the latter only certain sharing schemes are possible. Real valued configurations have been studied by Evans [10] and Gersht and Shulman [11] among others. Evans maximizes

a profit function, for example carried traffic, under a given maximum acceptable cell loss rate, and Gersht and Shulman maximize the minimum of residual link capacities under given service constraints.

The obvious way to obtain solutions for the capacitated VPCN design problem is to use constrained nonlinear programming (NLP) methods. However, for networks of realistic size, the number of decision variables and the computation times grow to an extent that this becomes impossible in practice. Herzberg and Byes [17] linearize the problem in order to reduce the computational burden and at the same time arrive at integer valued solutions using standard LP solvers, but do not touch upon the problem of a large state space. Gopal et al. [15, 16] and Arvidsson [1] devise heuristic optimization algorithms that maximize a function in a sequence of steps which converge to a (local) optimum. These algorithms belong to the class of so-called *greedy algorithms* where the action taken at each optimization step is the one which immediately gives the highest reward without considering long term impacts. Whereas Gopal et al. rely on predefined paths, Arvidsson includes path searching in his algorithm. Pióro and Gajowniczek [26] form and evaluate a large number of VPCN designs and select the best. This method is known as simulated allocation since each solution is computed by simulating capacity allocations and deallocations according to well known routing techniques.

2.4 The Technique of Mitra, Morrison and Ramakrishnan

Mitra et al. [25] emphasize that ATM network design and optimization at the call-level may be formulated within the framework of multirate, circuit-switched, loss networks with effective bandwidth encapsulating cell-level behaviour. Thus the model they use is very similar to the one presented in section 2.2. They employ several asymptotic results to reduce the complexity of the numerical calculations, the central one being the uniform asymptotic approximation (UAA) developed by Mitra and Morrison [24] for the anal-

ysis of single links. The UAA does not require separate treatment of the underloaded, critical or overloaded regimes. The complexity of calculating the blocking probabilities on a link carrying multirate traffic by the UAA is $O(S)$, that is, it is dependent only on S, the number of services, and not C, the number of circuits in the link. This contrasts with the well-known, exact calculation of the link blocking probabilities due to Kaufman [20] and Roberts [28] which has complexity $O(CS)$. Further, Mitra et al. give a unified approach which allows asymptotic and nonasymptotic (exact) calculation methods to be used cooperatively.

The problem considered is the calculation of the rates of traffic offered to various routes connecting origin–destination node pairs, which maximize network revenue. For calculating network loss probabilities Mitra et al. rely on the Erlang fixed point approximation, which is based on the link independence assumption, often referred to as the reduced load approximation, see Kelly [23]. The optimization procedure is guided by gradients obtained by solving a system of SL linear equations for the implied costs, where L is the number of links in the network, giving a complexity of $O(S^3L^3)$ for solving the equations. They show, however, that the UAA can be used to reduce the complexity to $O(L^3)$.

2.4.1 The Model

The model is similar to the one presented in section 2.2. Here, however, a traffic stream is characterized by a service class s and O-D pair (o, d), and has an admissible route set $\mathcal{R}^s_{o,d}$. The route sets $\{\mathcal{R}^s_{o,d}\}$ are assumed to be given and fixed. The outcome of a Bernoulli trial determines if an arriving call of a particular stream is offered to the network, which is a form of admission control, and, if it is admitted, to which route $r \in \mathcal{R}^s_{o,d}$ it is offered to. If an arriving call of class s is offered to route $r \in \mathcal{R}^s_{o,d}$, it will be blocked and lost if there is insufficient bandwidth available on any link of the route r. If a call is accepted, the bandwidth $d^s_{o,d}$ on each link is simultaneously held for the duration of the call.

The holding period of a call of class s connecting O-D pair (o,d) is generally distributed with mean $1/\mu^s_{o,d}$, and is independent of earlier arrival and holding times. The arrivals for a stream of class s connecting O-D pair (o,d) are Poisson, with mean rate $\lambda^s_{o,d}$, and arrivals of class s on route $r \in \mathcal{R}^s_{o,d}$ are Poisson, with mean rate λ^s_r, where

$$\sum_{r \in \mathcal{R}^s_{o,d}} \lambda^s_r \leq \lambda^s_{o,d}. \tag{2.5}$$

The corresponding traffic intensity is $\rho^s_r = \lambda^s_r/\mu^s_{o,d}$. It follows that

$$\sum_{r \in \mathcal{R}^s_{o,d}} \rho^s_r \leq \rho^s_{o,d} \tag{2.6}$$

where $\rho^s_{o,d} = \lambda^s_{o,d}/\mu^s_{o,d}$. The probability

$$p^s_{o,d} = \frac{1}{\lambda^s_{o,d}} \sum_{r \in \mathcal{R}^s_{o,d}} \lambda^s_r \tag{2.7}$$

is a parameter of the admission control, so that calls are offered to the network with probability $p^s_{o,d}$ and dropped with probability $(1-p^s_{o,d})$. Similarly, the probability that a call of service type s is offered to a route $r \in \mathcal{R}^s_{o,d}$ is $\lambda^s_r/\lambda^s_{o,d}$.

The important elements of the model are fixed route sets, a Bernoulli-trial admission control and state-independent call routing.

2.4.2 Network Design and Optimization

In line with the notation of section 2.2, let θ^s_r be the revenue earned per carried call per unit time by calls of service class s on route r, and let B^s_r be the equilibrium loss probability of service s on route $r \in \mathcal{R}^s_{o,d}$. The long-run average revenue rate for the network is

$$F = \sum_{o,d,s} \sum_{r \in \mathcal{R}^s_{o,d}} \theta^s_r \rho^s_r (1 - B^s_r). \tag{2.8}$$

The objective is to maximize the revenue F, subject to the constraints (2.6). The design variables are $\{\rho^s_r\}$. Note that the probabilities for admission control and routing $\{p^s_{o,d}\}$ are recovered from the design variables.

The optimization problem has a nonlinear objective function (2.8) and is subject to linear constraints (2.6). To evaluate F for given offered traffic rates $\{\rho_r^s\}$, it is necessary to calculate the loss probabilities $\{B_r^s\}$. The dependence of the loss probabilities on the offered traffic rates is complicated, which makes optimization hard. The optimization uses techniques described in following sections to obtain F and $\partial F/\partial \rho_r^s$. The steepest ascent direction is then obtained by projecting $\partial F/\partial \rho_r^s$ on to the null space of equation (2.6). This gives the direction in which to move. Next a search is made along this direction vector for an improved solution. The process is then iterated, until a local maximum of equation (2.8) is obtained. The fact that it is a local maximum implies that the complete process needs to be repeated using different initial conditions.

2.4.3 Network Analysis

This section presents the fixed point equations used for calculating the loss probabilities and the equations for the implied cost.

Fixed point equations

The fixed point equations use the reduced load approximation where each route which uses the link (o,d) offers an intensity to link (o,d), which is Poisson with a rate which is reduced by the independent thinning of all other traffics on the route. Let $\nu_{o,d}^s$ be the reduced load obtained after thinning of service class s calls offered to link (o,d), and let the bandwidth demand vector $\mathbf{d}_{o,d} = (d_{o,d}^1, \ldots, d_{o,d}^S)$, and $\boldsymbol{\nu}_{o,d} = (\nu_{o,d}^1, \ldots \nu_{o,d}^S)$. Then

$$B_{o,d}^s = E^s(\mathbf{d}_{o,d}, \boldsymbol{\nu}_{o,d}, D_{o,d}), \tag{2.9}$$

where the functions E^s may be calculated exactly by the recursive algorithm derived independently by Kaufman [20] and Roberts [28].

Using the reduced load approximation, the loads are given by

$$\nu_{o,d}^s = \sum_{o,d} \sum_{r \in \mathcal{A}_{o,d}^s} \rho_r^s \prod_{m \in r-(o,d)} (1 - B_m^s). \tag{2.10}$$

Here a route $r = \{(n_1, n_2), (n_2, n_3), \ldots, (n_{k-1}, n_k)\}$ is interpreted as a set of links and $r - (o, d)$ is a shorthand for $r \backslash \{(o, d)\}$.

The fixed point equations for the network may be summarized as follows

$$\mathbf{B} = \mathbf{\Phi}(\boldsymbol{\nu}) \tag{2.11}$$

$$\boldsymbol{\nu} = \mathbf{\Psi}(\mathbf{B}) \tag{2.12}$$

where $\boldsymbol{\nu} = (\nu^s_{o,d})_{o,d,s}$ and $\mathbf{B} = (B^s_{o,d})_{o,d,s}$; $\mathbf{\Phi}$ is obtained by fixing $\mathbf{d}_{o,d}$ and $D_{o,d}$ in (2.9) and $\mathbf{\Psi}$ is defined by equation (2.10).

The method of successive substitution is usually effective for solving the fixed point equations. This is done by starting with initial values for $\boldsymbol{\nu}$ and then alternating between calculating $\mathbf{B}$ from $\boldsymbol{\nu}$ using equation (2.11) and then calculating $\boldsymbol{\nu}$ from $\mathbf{B}$ using equation (2.12) until a convergence criterion is satisfied.

Under the link independence assumption, the loss probability B^s_r of service s on route r is

$$B^s_r = 1 - \prod_{m \in r} (1 - B^s_m), \quad \forall r \in \mathcal{R} \tag{2.13}$$

where $\mathcal{R} = \bigcup_{(o,d,s)} \mathcal{R}^s_{o,d}$. Note that $\{B^s_r\}$ is a distinct set of variables from $\{B^s_{o,d}\}$.

Implied costs

From equation (2.9) and the explicit expression for E^s (for example, see [20]), the following is obtained

$$\frac{\partial B^s_{o,d}}{\partial \nu^t_{o,d}} = (1 - B^t_{o,d})[E^s(\mathbf{d}_{o,d}, \boldsymbol{\nu}_{o,d}, D_{o,d} - d^t_{o,d}) - E^s(\mathbf{d}_{o,d}, \boldsymbol{\nu}_{o,d}, D_{o,d})]. \tag{2.14}$$

The revenue sensitivity to the offered loads, and the equations for the implied costs, may be obtained by extending the approach of Kelly [21] to the multirate case. This extension leads to the following result, when the revenue is calculated from the solution of the fixed point equations,

$$\frac{\partial F}{\partial \rho^s_r} = (1 - B^s_r)\left(e^s_r - \sum_{m \in r} c^s_m\right), \tag{2.15}$$

where the $\{c^s_{o,d}\}$ are the implied costs. Moreover, if $\nu^s_{o,d;r}$ denotes the thinned load of service class s on route r which is offered to link (o,d),

$$\nu^s_{o,d;r} = \rho^s_r \frac{(1-B^s_r)}{(1-B^s_{o,d})} = \rho^s_r \prod_{m \in r-(o,d)} (1-B^s_m), \tag{2.16}$$

then the implied costs satisfy the following system of SL linear equations,

$$c^t_{o,d} = \frac{1}{(1-B^t_{o,d})} \sum_{o,d,s} \sum_{r \in \mathcal{A}^s_{o,d}} \frac{\partial B^s_{o,d}}{\partial \nu^t_{o,d}} \nu^s_{o,d;r} \left(e^s_r - \sum_{k \in r-(o,d)} c^s_k \right). \tag{2.17}$$

The expression for $\partial B^s_{o,d}/\partial \nu^t_{o,d}$ given in equation (2.14) should be substituted in equation (2.17).

Note that the fixed point equations are independent of the equations for the implied costs, while the coefficients of the latter depend on the solution of the former. Specifically, $\{B^t_{o,d}\}$, $\{\partial B^s_{o,d}/\partial \nu^t_{o,d}\}$ and $\{\nu^s_{o,d;r}\}$ in equation (2.17) are obtained from the solution of the fixed point equations.

The computational complexity of solving the system of equations described by equation (2.17) is $O(S^3L^3)$.

2.4.4 Single Link Network Analysis

First the results from the exact analysis of a single link are given, and then the uniform asymptotic approximation (UAA) is applied to simplify the analysis. The UAA is applied in section 2.4.5 to networks with more than one link.

While considering a single link only, the subscripts for links, (o,d), and for routes, r, are suppressed. Here, traffic of service class s offered to the single link is ν^s.

Exact analysis

For a single link, the revenue sensitivities are given by

$$\frac{\partial F}{\partial \nu^s} = (1-B^s)(e^s - c^s), \tag{2.18}$$

where the loss probability of class s calls is

$$B^s = E^s(\mathbf{d}, \boldsymbol{\nu}, D), \quad s = 1, 2, \ldots, S, \tag{2.19}$$

and the implied costs are

$$c^t = \sum_{s=1}^{S} e^s \nu^s [E^s(\mathbf{d}, \boldsymbol{\nu}, D - d^t) - E^s(\mathbf{d}, \boldsymbol{\nu}, D)], \quad t = 1, 2, \ldots, S. \tag{2.20}$$

UAA analysis

We present a summary of the application of the UAA to the loss probabilities of a single link. The derivation was made by Mitra and Morrison [24] and assumes that

$$D \gg 1; \quad \nu^s = O(D), \quad s = 1, 2, \ldots, S, \tag{2.21}$$

where D is an integer. Also the d^s are positive integers, not large relative to D, and the greatest common divisor of $d^1, d^2, \ldots, d^S$ is assumed to be 1. Define

$$U(z) \equiv \sum_{s=1}^{S} \nu^s (z^{d^s} - 1) - D \log z \tag{2.22}$$

$$V(z) \equiv \sum_{s=1}^{S} (d^s)^2 \nu^s z^{d^s} \tag{2.23}$$

There is a unique positive solution z^* of $U'(z) = 0$, where the prime denotes derivative, so that

$$\sum_{s=1}^{S} d^s \nu^s (z^*)^{d^s} = D, \quad z^* > 0. \tag{2.24}$$

Since $U'(z)$ is strictly monotonic increasing with z, any simple procedure, such as bisection, quickly yields z^*.

Let

$$b^s = \begin{cases} \dfrac{1 - (z^*)^{d^s}}{1 - z^*}, & z^* \neq 1 \\ d^s, & z^* = 1 \end{cases} \tag{2.25}$$

and

$$B = \frac{\exp U(z^*)}{M\sqrt{2\pi V(z^*)}}, \tag{2.26}$$

where

$$M = \frac{1}{2}\operatorname{erfc}\left[\operatorname{sgn}(1-z^*)\sqrt{-U(z^*)}\right] \tag{2.27}$$

$$+ \frac{\exp U(z^*)}{\sqrt{2\pi}}\left[\frac{1}{\sqrt{V(z^*)}(1-z^*)} - \frac{\operatorname{sgn}(1-z^*)}{\sqrt{-2U(z^*)}}\right], \quad z^* \neq 1 \tag{2.28}$$

$$M = \frac{1}{2}\left\{1 + \frac{1}{\sqrt{2\pi V(1)}}\left[1 + \frac{1}{3V(1)}\sum_{s=1}^{S}(d^s)^3\nu^s\right]\right\}, \quad z^* = 1 \tag{2.29}$$

$$\tag{2.30}$$

and the complementary error function is given by

$$\operatorname{erfc}(x) = \frac{2}{\sqrt{\pi}}\int_x^\infty e^{-t^2}\,dt \tag{2.31}$$

Then, asymptotically,

$$B^s \sim b^s B, \quad s = 1, 2, \ldots, S. \tag{2.32}$$

These approximations hold whether $0 < z^* < 1$, $z^* \approx 1$ or $z^* > 1$, corresponding to the overloaded, critical and underloaded regimes, respectively. An analogous approximation to the derivative of B^s with respect to ν^t gives the following. Let

$$a^t = \frac{(z^*)^{d^t}}{(1-B^t)} - 1. \tag{2.33}$$

Then,

$$E^s(\mathbf{d}, \boldsymbol{\nu}, D - d^t) - E^s(\mathbf{d}, \boldsymbol{\nu}, D) = \frac{1}{(1-B^t)}\frac{\partial B^s}{\partial \nu^t} \sim a^t B^s. \tag{2.34}$$

Note that the complexity of calculating $\{B^s\}$ and $\{\partial B^s/\partial \nu^t\}$ is $O(S)$, i.e., bounded as $D \to \infty$.

Using equations (2.32) and (2.34) in equation (2.20), the asymptotic approximations of the implied costs for a single link are obtained,

$$c^t \sim a^t \xi, \tag{2.35}$$

where

$$\xi = \sum_{s=1}^{S} e^s \nu^s B^s \sim B \sum_{s=1}^{S} e^s \nu^s b^s. \tag{2.36}$$

Mitra et al. point out that in the overloaded regime, where $0 < z^* < 1$, it is the case that $B^t \sim 1 - (z^*)^{d^t}$, so that $a^t \sim 0$, and it is necessary to refine the approximation. They note that this may be done, but that the overloaded regime is not of great interest.

2.4.5 Network Analysis Based on UAA

With the foundations of sections 2.4.3 and 2.4.4, it is straightforward to describe the fixed point equations and the equations for the implied costs which are obtained from UAA, for the full network.

Fixed point equations

The fixed point equations for the network are given in equations (2.11) and (2.12). Under the UAA, the function $\mathbf{\Psi}$ is unchanged and is defined by equation (2.10). The function $\mathbf{\Phi}$ is quite different under the UAA. Effectively, this procedure describes a replacement for $\mathbf{\Phi}$ in equation (2.9). The procedure relies on the functions, with link subscripts added, $U_{o,d}(z)$ and $V_{o,d}(z)$ which are defined by equation (2.22) and (2.23). The point $z^*_{o,d}$ where the minimum $U_{o,d}(z^*_{o,d})$ occurs is calculated, $b^s_{o,d}$ and $B_{o,d}$ are calculated as in equations (2.25) and (2.26), and finally

$$B^s_{o,d} = b^s_{o,d} B_{o,d}.$$

Once again, the technique of successive substitution previously described in section 2.4.3 may be used to solve the equations.

Implied costs

Equation (2.17) gives the system of linear equations satisfied by the implied costs. Equation (2.34) obtained by UAA is substituted into equation (2.17),

$$\frac{1}{(1-B^t_{o,d})}\frac{\partial B^s_{o,d}}{\partial \nu^t_{o,d}} = a^t_{o,d} B^s_{o,d} \tag{2.37}$$

where

$$a^t_{o,d} = \frac{(z^*_{o,d})^{d^t_{o,d}}}{(1-B^t_{o,d})} - 1. \tag{2.38}$$

This gives

$$c^t_{o,d} = a^t_{o,d} \sum_{o,d,s} \sum_{r \in \mathcal{A}^s_{o,d}} \nu^s_{o,d;r} B^s_{o,d} \left(e^s_r - \sum_{m \in r-(o,d)} c^s_m \right). \tag{2.39}$$

Now define

$$\xi_{o,d} \equiv \sum_{o,d,s} \sum_{r \in \mathcal{A}^s_{o,d}} \nu^s_{o,d;r} B^s_{o,d} \left(e^s_r - \sum_{m \in r-(o,d)} c^s_m \right). \tag{2.40}$$

Note that for all t and (o,d),

$$c^t_{o,d} = a^t_{o,d}\, \xi_{o,d}. \tag{2.41}$$

Now introduce equation (2.41) into equation (2.40) to obtain

$$\xi_{o,d} = \sum_{o,d,s} \sum_{r \in \mathcal{A}^s_{o,d}} \nu^s_{o,d;r} B^s_{o,d} \left(e^s_r - \sum_{m \in r-(o,d)} a^s_m \xi_m \right). \tag{2.42}$$

Equation (2.42) is a complete system of equations in $\{\xi_{o,d}\}$. The parameters in this system of equations, namely, $\{\nu^s_{o,d;r}\}$, $\{B^s_{o,d}\}$ and $\{a^s_m\}$ are all obtained directly from the solution of the fixed point equations in section 2.4.5. Once the solution of equation (2.42) has been obtained, all the implied costs $\{c^t_{o,d}\}$ are obtained from equation (2.41).

Chapter 3

XFG: An Optimization Algorithm

This chapter describes the XFG algorithm which efficiently determines an optimal virtual path connection network (VPCN) configuration, given a physical network and a traffic intensity matrix. We present the algorithm in its standard form, with neither consideration for the rapid calculation of a series of optimal VPCNs (as would be calculated in response to a series of changing traffic matrices) nor for the specific implementation details of XFG.

3.1 General Approach

The XFG algorithm computes the optimal solution to the VPCN design problem described in section 2.2 of chapter 2. Recall that the design problem is to find the configuration $\mathbf{x}_{\text{opt}}$ that maximizes the revenue earning rate

$$\begin{aligned} F(\mathbf{x}) &= \sum_{o,d} F_{o,d}(C_{o,d}(\mathbf{x})) \\ &= \sum_{o,d,s} \theta^s_{o,d}\, \rho^s_{o,d}(1 - E^s_{o,d}(C_{o,d}(\mathbf{x}))) \end{aligned} \tag{3.1}$$

subject to the constraints

$$x_r \geq 0 \tag{3.2}$$

for all $r \in \mathcal{R}$, and for all (o, d)

$$\sum_{r \in \mathcal{A}_{o,d}} x_r \leq D_{o,d} \quad . \tag{3.3}$$

The XFG algorithm is in the class of greedy optimization algorithms, but with the significant difference that XFG computes a global optimum.

The XFG algorithm is based on the concept of link costs. When the system configuration is $\mathbf{x}$, the cost $q_{o,d}(c)$, $c = C(\mathbf{x})$ of the link or VPC (o, d) is defined as the derivative of the link revenue function with respect to the capacity allocation

$$q_{o,d}(c) = \frac{\partial F_{o,d}(c)}{\partial c} \tag{3.4}$$

at $c = C_{o,d}(\mathbf{x}) > 0$. Furthermore, the algorithm is designed to avoid *irregular* solutions. An irregular solution is a configuration $\mathbf{x}$ where the entire capacity of a physical link (o, d) is allocated to routes $r \in \mathcal{A}_{o,d}$ so that

$$\sum_{r \in \mathcal{A}_{o,d}} x_r = D_{o,d}. \tag{3.5}$$

In such a case, none of the offered traffic $\rho_{o,d}$ is carried on the physical link (o, d) and the capacity $C_{o,d}(\mathbf{x})$ assigned to carry the offered traffic $\rho_{o,d}$ is composed entirely of routes

$$C_{o,d}(\mathbf{x}) = \sum_{r \in \mathcal{R}_{o,d}} x_r. \tag{3.6}$$

Such configurations can be avoided by setting the cost of a link from which all direct traffic has been displaced to an overall network maximum so that for all o and d

$$q_{o,d}(c) = \theta_{\max} = \max_{o,d,s}(\theta^s_{o,d}/d^s) \tag{3.7}$$

at $c = C_{o,d}(\mathbf{x}) = 0$ where $\theta_{\max}$ is the maximal rate of earning revenue per unit of capacity. Equations (3.4) and (3.7) fully describe the link cost function

$$q_{o,d}(c) = \begin{cases} \dfrac{\partial F_{o,d}(c)}{\partial c}, & c > 0 \\ \theta_{\max}, & c = 0\,. \end{cases} \tag{3.8}$$

The cost $q_r(\mathbf{x})$ of a route $r = (n_1, n_2, \ldots, n_k)$ is the sum of the costs of the links comprising the route

$$q_r(\mathbf{x}) = \sum_{\ell=1}^{k-1} q_{n_\ell, n_{\ell+1}}(C_{n_\ell, n_{\ell+1}}(\mathbf{x})). \tag{3.9}$$

Since the XFG algorithm is based on balancing link costs, the requirement expressed in equation (3.7) means that the situation where a link is entirely used by non-direct traffic will not occur in an optimal solution, hence the physical constraints given in equations (3.2) and (3.3) are taken care of implicitly.

The following factors are taken into account when computing the link costs. Firstly, the link revenue function $F_{o,d}(c)$ given in equation (3.1) is defined for integral values of the link capacity c. The link cost is approximated by equating it to the loss in revenue earned from from carrying the traffic (o, d) on the VPC (o, d) when the capacity of the VPC (o, d) is decreased from c to $c - 1$ or, equivalently, the increase in revenue when the capacity of the VPC is increased from c to $c + 1$.

Secondly, most optimization techniques require that the function $F(c)$ being optimized be a smooth, monotone and convex function for all real nonnegative c. Since an NLP code was used to confirm (for small networks) the optimality of the XFG computed VPCN configurations [5], the link revenue function $F_{o,d}(c)$ is chosen to be monotone and convex, although the XFG algorithm itself does not require this condition. However, if $F_{o,d}(c)$ is not monotone and convex then the XFG algorithm will not necessarily converge to a global optimum.

The rate $F^s_{o,d}(c)$ at which calls of service s between nodes o and d earn revenue is a non-monotonic function of the capacity c of the VPC connecting nodes o and d [30]. However, in most cases of practical interest, the link revenue function $F_{o,d}(c) = \sum_{s=1}^{S} F^s_{o,d}(c)$ is very nearly monotone and convex. Several arguments support this statement: VPC capacities in ATM networks are in general much larger than the largest bandwidth requirement d^S and therefore the influence of the multiservice factor which gives rise to the non-monotonicity is limited; if the revenue is proportional to the bandwidth, or if

d^S is chosen as the unit of increment of VPC capacity, then the link revenue function is monotone; the link revenue function will be monotone if calls are accepted only if there are at least d^S circuits available on the VPC – this will result in equal loss probabilities for all services which in turn suggests that the multiservice loss probabilities will behave in a similar manner to single-service loss probabilities.

Therefore, instead of optimizing $F(\mathbf{x})$, we optimize the function $\tilde{F}(\mathbf{x}) = \sum_{o,d} \tilde{F}_{o,d}(C_{o,d}(\mathbf{x}))$ where $\tilde{F}_{o,d}(c)$ is obtained by fitting a differentiable increasing convex function to $F_{o,d}(c)$ where $c = C_{o,d}(\mathbf{x})$. Since $F_{o,d}(c)$ is nearly monotone and convex, the difference $\tilde{F}_{o,d}(c) - F_{o,d}(c)$, which is the error in the approximation, will be small and will be zero if $F_{o,d}(c)$ is monotone and convex.

For the remainder of the chapter, since no confusion can arise, let $F(\mathbf{x})$ denote $\tilde{F}(\mathbf{x})$. Note that $F(\mathbf{x})$ is convex since (i) $F(\mathbf{x})$ is the sum of convex functions and (ii) if $\mathbf{x}'$ and $\mathbf{x}''$ are feasible solutions then their linear combination $\alpha\mathbf{x}' + (1-\alpha)\mathbf{x}''$ is a feasible solution too, where $0 \leq \alpha \leq 1$. Note that (ii) implies that if $\mathbf{x}'$ and $\mathbf{x}''$ are local maximums of $F(\mathbf{x})$ then $F(\mathbf{x}') = F(\mathbf{x}'') = F_{\max}$.

The necessary and sufficient condition for $\mathbf{x}$ to be an optimum for $F(\mathbf{x})$ is that for any O-D pair (o,d) and for any route $r \in \mathcal{R}_{o,d}$ such that $x_r > 0$

$$q_{o,d}(C_{o,d}(\mathbf{x})) = q_r(\mathbf{x}) \tag{3.10}$$

and for any O-D pair (o,d) such that $\mathcal{R}_{o,d} = \emptyset$

$$q_{o,d}(C_{o,d}(\mathbf{x})) \leq 0 \tag{3.11}$$

The configuration $\mathbf{x}$ is thus optimal when the revenue gained by adding more capacity to any VPC (o,d) is less than the cost incurred in acquiring this capacity from the links of the route $r \in \mathcal{R}_{o,d}$; and for any route $r \in \mathcal{R}_{o,d}$ the revenue lost by releasing capacity from the VPC (o,d) exceeds the revenue gained when this capacity is returned to the links of the route r.

The XFG algorithm can now be given. The basic principle can be compared to a bandwidth market where only one transaction at a time is carried out

before demands and supplies are reevaluated and prices adjusted accordingly. A transaction can either be a route buying capacity from a set of links (allocation) or a set of links buying capacity from a route (deallocation).

3.2 The Algorithm

This section describes the XFG algorithm for finding an optimal VPCN.

1. **Initialization:** Set $n := 0$ and construct an arbitrary initial configuration $\mathbf{x}_0$. For example, an initial configuration may have no routes so that $\mathcal{R} = \emptyset$ and the VPCs each span one link, or the initial configuration may estimate the optimal configuration.

2. **Compute the least-cost paths:** Construct a weighted graph of the physical network where each VPC is assigned its current cost. For each O-D pair (o, d) use Dijkstra's algorithm [7] to find the least-cost routes r between o and d.

3. **Find the most attractive allocation:** Select the O-D pair (o^+, d^+) for which the difference δ_{r^+} between the cost of VPC (o^+, d^+) and the cost of the least-cost route r^+ connecting o^+ to d^+ is maximal, where the costs are given by equations (3.8) and (3.9) respectively,

$$\delta_{r^+} = q_{o^+,d^+} - q_{r^+}.$$

4. **Find the most attractive deallocation:** Consider all routes r with $x_r > 0$. Select the route $r^- \in \mathcal{R}_{o^-,d^-}$ for which the difference δ_{r^-} between the cost of route r^- and the cost of the VPC (o^-, d^-) is maximal, where the costs are given by equations (3.9) and (3.8) respectively,

$$\delta_{r^-} = q_{r^-} - q_{o^-,d^-}.$$

5. **Convergence test:** If both δ_{r^+} and δ_{r^-} are negative or, up to a very small quantity, equal to zero, then $F(\mathbf{x}_n)$ is, to the prescribed level of accuracy, an optimum and the algorithm terminates.

6. **Perform the most attractive transaction:** Obtain a configuration $\mathbf{x}_{n+1}$ which yields a larger revenue than $\mathbf{x}_n$ by performing either an allocation or a deallocation, depending upon which transaction is the most profitable.

   Allocation: If $\delta_{r+} > \delta_{r-}$ then allocate δx^+ units of capacity to the route r^+

   $$x_{r+} := x_{r+} + \delta x^+.$$

   Add the route r^+ to the route set

   $$\mathcal{R}_{o^+,d^+} := \mathcal{R}_{o^+,d^+} \cup r^+$$

   and for all links (i,j) in the route r^+

   $$\mathcal{A}_{i,j} := \mathcal{A}_{i,j} \cup r^+.$$

   Reevaluate the capacities of the VPC (o^+, d^+) and the VPCs (i,j) in r^+ according to equation 2.1.

   Deallocation: Else if $\delta_{r-} > \delta_{r+}$ then deallocate δx^- units of capacity to the route r^-

   $$x_{r-} := x_{r-} - \delta x^-.$$

   If $x_{r-} = 0$ remove the route r^- from the route set

   $$\mathcal{R}_{o^-,d^-} := \mathcal{R}_{o^-,d^-} \backslash r^-$$

   and for all links (i,j) in the route r^-

   $$\mathcal{A}_{i,j} := \mathcal{A}_{i,j} \backslash r^-.$$

   Reevaluate the capacities of the VPC (o^-, d^-) and the VPCs (i,j) in r^- according to equation 2.1.

7. **Loop statement:** $n := n + 1$ and go to step 2.

The amounts of capacity δx^+ and δx^- that are allocated or deallocated at each step can be found so that they maximize the increase in revenue (line

search). If the $\delta x^{\pm}$ are real-valued, the the XFG algorithm will compute a globally optimal solution $\mathbf{x}$ where the route capacities x_r are real valued. If the $\delta x^{\pm}$ are nonnegative integers, then the algorithm will compute integer-valued route capacities. In this case the final configuration $\mathbf{x}$ need not be a global optimum, but $\mathbf{x}$ will be close to an optimum.

3.3 XFG and Other Solvers

Several difficulties are encountered when standard NLP solvers are used to solve the VPCN design problem. These difficulties arise from the fact that standard nonlinear programme (NLP) solvers make little user of the network structure that is implicit in the function being maximized, equation (3.1). In contrast, the XFG optimization method is based entirely on using the structure of the network, and on the form of the link revenue function so that XFG is more efficient than standard NLP solvers when applied to the VPCN design problem.

3.3.1 The Size of the Problem

$F(\mathbf{x})$ is a function of $R = |\mathcal{R}|$ variables corresponding to the total number of routes in the network. If all possible routes are included, R becomes enormous and this is so even if limitations are imposed on the number of routes to be considered. For example, let $\mathcal{R}^0_{o,d}$ denote the set of shortest routes between nodes o and d and let $L^0_{o,d}$ denote the length of these routes. Let $\mathcal{R}^m_{o,d}$ denote the set of routes between nodes o and d of length $L^m_{o,d} = L^0_{o,d} + m$ where m is a positive integer. A natural choice of decision variables would be to work with the routes $r \in \sum_{\ell=0}^{m} \bigcup_{(o,d)} \mathcal{R}^{\ell}_{o,d}$. However, even for a small network with a few dozen nodes and for a small value of m, this choice will result in a set of variables that is so large that the optimization becomes virtually impossible, even for efficient NLP solvers.

The XFG algorithm, on the other hand, considers *all* possible routes, even in very large networks. It does so, not by precomputing all possible routes,

but by exploring the network for the best routes at each iteration of the algorithm. In this way, XFG identifies and works with *active* routes, namely those routes for which $x_r > 0$. Experiments have shown that the number of active routes is much smaller than the total number of possible routes.

3.3.2 Gradient Moves

All NLP solvers, including XFG, use some strategy to move from a configuration $\mathbf{x}$ to a configuration $\mathbf{x}'$ with improved revenue. Most standard NLP solvers use the projection of the gradient on the space defined by the linear constraints in their search for the next configuration. This has several consequences. Firstly, calculating the projection is computationally expensive. Secondly, moving along this direction implies changing most of the components of the vector $\mathbf{x}$, which requires substantial processing time. Thirdly, computing the length of the move in the selected direction (using line search) also takes substantial computational resources. Finally, the gradient is calculated with respect to the Euclidean distance between the configurations, whereas the network structure implicit in the VPCN design problem hints at using another definition of distance.

The strategy used by XFG to move to an improved configuration is in some ways better than the gradient projection move. This is shown here informally, without employing mathematically strict constructions, the intention being to justify expectations about the properties of the XFG solver.

The link revenue function in equation (3.1) may be viewed as a function of the variables $\{C_{o,d}\}$ which are in turn related via the variables $\{x_r\}$ and the constraints given in equations (3.2) and (3.3). Standard NLP solvers use the gradient as the direction of the best unit move in $\{x_r\}$.

However, the real interest lies in the best unit move in $\{C_{o,d}\}$ provided that there exists a change in $\{x_r\}$ which results in a corresponding change in $\{C_{o,d}\}$. When choosing the best unit direction $\delta\mathbf{C}$, a different distance

between the configurations might be considered, namely

$$\sum_{o,d} |\delta C_{o,d}|$$

instead of the usual

$$\sqrt{\sum_{o,d} \delta C_{o,d}^2}.$$

For example, consider the two configurations $\mathbf{x} = (0,0)$ and $\mathbf{x}' = (1,1)$. The distance between $\mathbf{x}$ and $\mathbf{x}'$ should rather be defined as 2 since two units of capacity are allocated in order to move from $\mathbf{x}$ to $\mathbf{x}'$ and not $\sqrt{2}$. If we use such a norm for vectors, then instead of increasing (decreasing) capacities along several links at a time we will increase (decrease) capacity along the route which gives the highest revenue increase. This will be the gradient, which is the most profitable unit move with respect to the defined norm and this is precisely what XFG does.

Admittedly that this is not a strictly accurate statement. Indeed, there are some difficulties, namely the relationship between $\{C_{o,d}\}$ and $\{x_r\}$ via the constraints given in equations (3.2) and (3.3). However, these arguments suggest that the improvement directions used by XFG are good in principle. Moreover, our experiments confirm that when applied to small networks, XFG is several orders of magnitude faster than other NLP solvers. XFG changes only one variable at a time. Each such change will alter the capacities of only a few links, and in most cases by one unit. If XFG uses a line search it will make comparisons for only a few links.

3.3.3 Regular Configurations

Standard NLP solvers compute a sequence of configurations where each successive configuration earns an improved revenue. However, most of these configurations will be irregular. The XFG solver works only with regular configurations. Substantially less computational effort is required if only regular configurations are considered. As indicated above, it is easy to show that a regular optimal solution exists and that regular solutions make sense.

An irregular solution, on the other hand, will contain links which are simultaneously both under-configured (they require extra capacity) and over-configured (they contribute capacity to other routes) and this makes little sense.

Regularity is important because the restriction to regular configurations dramatically reduces the solution state space – there are many irregular solutions that correspond to the same regular solution.

3.3.4 Integer versus Real-valued Solutions

NLP solvers will, in general, compute a solution $\mathbf{x}$ with route capacities $\{x_r\}$ that are real valued. Our implementation of the XFG algorithm computes solutions with integer valued route capacities. Our experiments confirm that the integer-valued implementation of the XFG algorithm produces, with modest programming effort, an excellent suboptimal solution. Small networks optimized by other methods including standard NLP packages and simulated annealing, have not found superior optima for these networks. Nonetheless, the integer-valued implementation of the XFG algorithm cannot be guaranteed to avoid inferior local optima.

The XFG algorithm, when working with real values, computes an optimal solution $\mathbf{x}_{\text{opt}}$ which is independent of the choice of the initial configuration $\mathbf{x}_0$. Experiments indicate that this property is retained for the integer valued implementation of the XFG algorithm where $\mathbf{x}_{\text{opt}}$ is almost independent of $\mathbf{x}_0$. Furthermore, commencing the XFG optimization from an initial configuration $\mathbf{x}_0 \sim \mathbf{x}_{\text{opt}}$ will substantially reduce the computation required to calculate the optimal configuration $\mathbf{x}_{\text{opt}}$.

3.3.5 Other Optimization Techniques

XFG compares well with other optimization techniques that take the structure of the network into account. By working with a fixed VPCN, XFG uses an exact model of the network, rather than a fixed-point, or reduced load,

approximation which often provides a poor approximation for multiservice networks, see Ross [30, chapter 8]. For example, the technique of Mitra et al. described in chapter 2 use the fixed-point approximation approach.

The technique of Mitra et al. is an example of where several approximations are used. In their case, the uniform asymptotic approximation for calculating link blocking probabilities is used. XFG uses a numerically stable variant of the efficient recursive calculation method of Roberts [28] and Kaufman [20] to calculate the link blocking probabilities which are then stored in a lookup table. This is possible because XFG uses a fixed VPCN model, and provides XFG with a means of obtaining the link blocking probabilities independently of the size of the link.

Finally, XFG's simplicity makes it attractive. When compared with the technique of Bauschert [3], for example, which uses a multi-level algorithm, XFG is less prone to implementation error.

Chapter 4

XFG Internals

This chapter describes the features of XFG that are specific to its computer implementation including options to speed up the algorithm (at the expense of some accuracy), internal techniques for improving efficiency and avoiding undesirable configurations, and the ability to initialize the computation from a previous configuration.

4.1 Checkpointing

The checkpointing feature of the XFG implementation makes it possible for the XFG algorithm to be used for dynamic reconfiguration of networks in near-realtime.

Checkpointing is XFG's ability to read and write checkpoint files which fully describe the VPCN configuration, including all virtual paths, all routes, the donor and recipient status of links and the expected rate of revenue in that configuration. Link (o, d) is a donor if it contributes circuits to a route, which is the case when $\mathcal{A}_{o,d}$ is non-empty; and link (o, d) is a recipient if the virtual path (o, d) is composed of one or more routes, that is $\mathcal{R}_{o,d}$ is non-empty.

XFG generates checkpoint files as follows. When the XFG algorithm terminates, enough information is written from the data structures in memory to

the file to reconstruct those structures which describe the optimal VPCN. Information obtained from the XFG input file, including the physical network structure and the arrival rates, is not stored in the checkpoint file, as this information might change from one invocation of XFG to another.

When XFG reads an initial configuration from a checkpoint file, it does not simply read the network description and immediately proceed to the optimization cycle. Instead it first performs a "backoff" operation where it releases circuits from routes on overcapacitated virtual paths and routes that use undercapacitated links. Such VPCs were optimally configured when the checkpoint file was written. However, the network link capacities and offered traffic may have changed so that, when the checkpoint file is read, the previously optimal VPCs are no longer optimal.

Releasing circuits from overcapacitated VPs is accomplished by repeatedly searching for a route where a release of capacity will afford the best increase in the rate of earning revenue and then making the release. The same technique is used as in step 4 of the XFG algorithm (see page 27). The process is repeated until no route exists where a further release of capacity would afford an increase in the rate of earning revenue.

The process of releasing capacity on routes that use undercapacitated links does not have a counterpart step in the XFG algorithm. All links which are donors (that is, their circuits are used to carry non-direct traffic) are considered in turn. For a given donor link, all routes which use circuits from the link are searched for the one whose release will afford the best increase in rate of earning revenue. If a revenue increasing release exists, then the best release is made, else the next donor link is considered.

Our experiments show that the backoff procedure is useful for finding a good starting configuration $\mathbf{x}_0$ for the integer valued implementation of XFG. Recall that for the real valued implementation, any initial configuration $\mathbf{x}_0$ may be used. The integer valued implementation, however, can run into problems when initialized from a previously optimal configuration. In addition, the backoff procedure rapidly moves away from the (now) sub-optimal

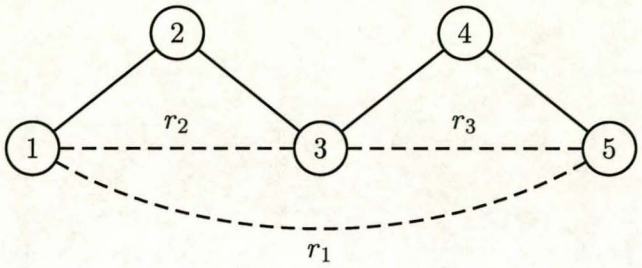

Figure 4.1: Failure of the integer valued XFG implementation

configuration obtained from a previous optimization, making it more attractive for the computer implementation of XFG. The backoff procedure also ensures that the initial configuration does not violate any of the constraints imposed by the physical network, should it have changed.

4.2 A Boundary Condition

In most cases of practical importance, the integer valued implementation of the XFG algorithm will produce, with modest programming and computational effort, an excellent suboptimal solution which is very close to the optimum. However, in some cases it can produce a solution which is not optimal. The problem arises if the derivative of the revenue function $q_{o,d}(c)$ is not modified at $c = 0$ for O-D pairs (o, d) which have circuits, $D_{o,d} > 0$, but to which no direct traffic is offered, $\rho_{o,d} = 0$ – see equation (3.7). The following example demonstrates the problem.

Consider the single-rate network presented in figure 4.1. Let the traffic intensities be $\rho_{1,3} = \rho_{3,5} = \rho_{1,5} = 1000$. Let the revenues be $\theta_{1,3} = \theta_{3,5} = 1$ and $\theta_{1,5} = 1.1$. The link sizes are $D_{1,2} = D_{2,3} = D_{3,4} = D_{4,5} = 1000$. No traffic is offered to the physical links. The costs – in the sense of equation (3.4) – of the physical links are therefore zero: XFG will assign circuits on route $r_1 = (1, 2, 3, 4, 5)$ most of the time, since the revenue generated by traffic on route r_1 is higher than the revenue generated by traffic on routes $r_2 = (1, 2, 3)$ and $r_3 = (3, 4, 5)$. Note that we use the term *circuit* and *capacity* synonymously: one circuit is equivalent to one unit of bandwidth. As

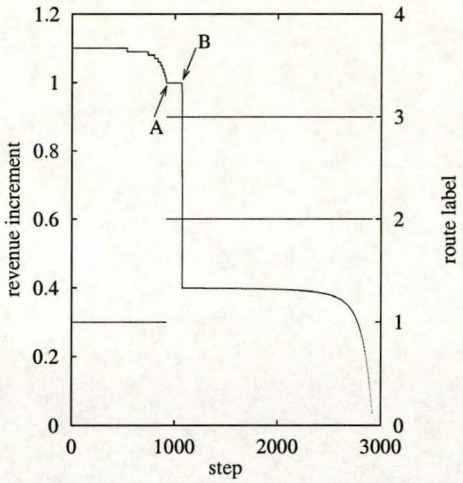

Figure 4.2: Route selection and incremental revenue earned per connection

a result of these assignments, the capacity of the physical links will be fully reserved for the use of the traffic on routes r_1, r_2 and r_3 with 921 circuits being reserved for route r_1 and 78 circuits being reserved for each of the routes r_2 and r_3. This solution is far from optimal, since it generates 1161 units of revenue, whereas 1951 units of revenue would be generated if all the capacity were given to routes r_2 and r_3. XFG will not make any releases because no cost is associated with the physical links (no traffic is offered to the physical links) and the algorithm cannot decide which traffic should better use the existing capacity.

The XFG implementation is easily adapted to resolve this problem. The requirement that $q_{o,d}(0) = \theta$ as specified in the XFG algorithm (see equation (3.7), chapter 3) is approximately equivalent to offering a small amount of direct traffic to each physical link. The link cost thus rises steeply when nearly all of the capacity of the link is assigned to routes that use the link. Figure 4.2 shows the revenue increment

$$\Delta F(\mathbf{x}_n) = F(\mathbf{x}_n) - F(\mathbf{x}_{n-1}) \tag{4.1}$$

gained by assigning one circuit at step n, and the route selected at step n. The algorithm initially assigns 921 circuits along route r_1, until 79 circuits remain on each of the physical links – see the point labelled "A" in figure 4.2.

The costs of the physical links are now sufficiently high to prevent XFG from allocating more circuits along route r_1. The algorithm next assigns 78 circuits along each of the routes r_2 and r_3 until one circuit remains on each of the physical links – see the point labelled "B" on figure 4.2. The cost of the physical links is now so high that route r_1 releases one circuit to each of the physical links. This is immediately followed by two assignments: one circuit is assigned to route r_2 and then one circuit is assigned to route r_3. As a result, the cost of the physical links rises again and another release on route r_1 follows. XFG will release, one by one, all of the 921 circuits assigned on route r_1 and assign the freed capacity to the routes r_2 and r_3. The algorithm terminates when 999 circuits are reserved for routes r_2 and r_3. The XFG algorithm produces a near optimal revenue of 1951 – the optimal revenue is produced when 1000 units of capacity are reserved for routes r_2 and r_3.

4.3 Lookup Tables

Before describing how XFG uses lookup tables to gain a speed advantage, we first describe the format used for inputting the call traffic arrival intensities and rates of revenue. Let $\rho^s_{o,d}$ denote the intensity of a traffic stream of class s offered to the origin–destination pair (o, d). The computer implementation of XFG uses a matrix $\mathbf{\Gamma} = [\gamma_{o,d}]$ and a vector $\mathbf{K} = [k^s]$ to describe the $\{\rho^s_{o,d}\}$,

$$\rho^s_{o,d} = k^s \gamma_{o,d}. \tag{4.2}$$

This description substantially reduces the amount of storage required to represent the class dependent traffic intensities. The $\{\gamma_{o,d}\}$ represent the call traffic intensities between O-D pairs, while the $\{k^s\}$ give the load factors that differentiate the call classes. The computer implementation of XFG uses a similar technique with the rates of earning revenue: the matrix $\mathbf{T} = [\tau_{o,d}]$ gives the rate of earning per bandwidth unit for O-D pair (o, d) so that

$$\theta^s_{o,d} = d^s \tau_{o,d}. \tag{4.3}$$

XFG uses lookup tables to store the values of the functions $\{F_{o,d}\}$. Recall

that the link revenue function

$$\begin{aligned} F_{o,d}(c) &= \sum_{s=1}^{S} \theta^s_{o,d}\, \rho^s_{o,d}(1 - E^s_{o,d}(c)) \\ &= \sum_{s=1}^{S} d^s \tau_{o,d}\, k^s \gamma_{o,d}(1 - E^s_{o,d}(c)). \end{aligned} \tag{4.4}$$

The rate of earning revenue in a given configuration $\mathbf{x}$ on a particular O-D pair (o, d) is $F_{o,d}(C_{o,d}(\mathbf{x}))$.

To limit the size of the lookup tables, XFG calculates $C_{\max}$, the maximum number of circuits required to carry call traffic with a grade of service of $B = 0.001$. This is an order of magnitude less than typical GOS values in the networks XFG is used to optimize. For networks where extremely good (that is, small) GOS values are required, this internal value may be set to find the best trade-off between accuracy and storage space. The number of circuits required to carry call traffic with a given blocking probability is calculated using a bound [6] on the inverse of Erlang's function,

$$C(\rho, B) < \rho(1 - B) + 1/B. \tag{4.5}$$

Because the inverse of Erlang's function is for single-rate traffic, the value of ρ in equation (4.5) is approximated as the weighted sum of arrival intensities of all service classes, $\rho = \sum_s d^s\, \rho^s_{o,d}$. The value of $C_{\max}$ is calculated as

$$C_{\max} = \max_{o,d} \left[\left(\textstyle\sum_s k^s \gamma_{o,d}\, d^s \right) (1 - B) + 1/B \right]. \tag{4.6}$$

Once $C_{\max}$ has been calculated, the rate of revenue $F_{o,d}(c)$ is calculated for each O-D pair (o, d) for all integer values of c from 1 to $C_{\max}$ and stored in a vector. If successive values of $F_{o,d}(c)$ differ by insignificant amounts, the vector is truncated, and the value of c at this point recorded as an effective $C_{\max}$ for that O-D pair. A convex hull is fitted to the function described by the vector, and this hull becomes the lookup table used to evaluate the rate of earning revenue during the optimization process. Only enough points necessary to recover the convex hull are stored, and the link revenue rates are determined using either linear or cubic spline interpolation.

A revenue table is created for each O-D pair, except that O-D pairs which have the same arrival intensity $\gamma_{o,d}$ and the same revenue rate $\tau_{o,d}$ share a

single copy of a revenue table. Table sharing can substantially reduce the storage requirements of XFG.

Table sharing is especially useful when evaluating prototype networks during the design of a new network. In such cases, as an initial approximation, the origin–destination pairs may be grouped into a few classes and the call arrival intensities for members of a class may be set to the same value. This has the obvious advantages of reduced storage space requirements and reduced startup computation time.

4.4 Allocation Unit and Other Tuning Parameters

The computer implementation of the XFG algorithm provides several parameters which may be tuned by the user. The most important two are the prerelease and postrelease allocation units.

The initial (often many thousand) steps of the XFG algorithm discover routes and allocate capacity on these routes. This is necessary, because except in the case of a physically fully connected network where the physical links have close to enough circuits to carry the offered traffic, routes need to be created to carry traffic where no direct link exists. Only after many steps does the algorithm start to trade routes for more profitable ones. How long it takes before the first route deallocation occurs, varies greatly from network to network and is dependent on both the physical network structure and its traffic demand characteristics.

By default XFG allocates circuits to routes one circuit at a time. (We use the term *circuit* and *capacity* synonymously so that one circuit is equivalent to one bandwidth unit.) By specifying a *prerelease allocation unit,* circuits may be allocated in larger groups. This considerably improves the speed of XFG by reducing the number of steps taken before the first deallocation is made. Once the first deallocation is made, XFG uses a *postrelease allocation unit* which is usually set to one circuit. The finer grained movements of a smaller allocation unit are more important once deallocations begin, and

this helps to avoid inferior local optima by making the integer valued XFG implementation a better approximation of the real valued algorithm.

The *start release parameter* is related to the prerelease and postrelease allocation units. If set, XFG will not start testing for possible release candidates until the rate of revenue increment per step drops below this value. This parameter was introduced in order to avoid the unnecessary computation involved in testing for release candidates during the initial stage when only connections are made. The XFG algorithm is sensitive to the setting of this parameter's value. For example, if release checking is suppressed by assigning a very small value to the start release parameter, then checking for release candidates will start late, only after such candidates become available. On the other hand, if checking for release candidates starts too soon, then much CPU time is wasted testing for unavailable release candidates. Furthermore, if a prerelease allocation unit is used in conjunction with an inappropriately small start release parameter, then XFG is very likely to calculate a suboptimal configuration.

In practice no simple heuristic could be found to set the start release parameter with any reliability. A trial and error technique was useful for finding a suitable value when many XFG runs were performed during the testing of the XFG implementation. However, most of the investigations with XFG were performed without setting this parameter.

A further tuning parameter that may be set is the *convergence criterion.* XFG terminates when the increase in revenue from one step to the next is less than the value of this parameter. This parameter is useful where the rates of earning revenue are numerically large and the default convergence test of 10^{-6} is inappropriate. The parameter was also used to examine the computational savings that could be obtained at the expense of a less accurate solution.

The computer implementation of XFG allows the specification of the *maximum route length* that is to be considered. This is useful when comparing XFG with other optimization algorithms where it is not possible to consider

routes of arbitrary length. By running XFG once with this parameter set and once without, an estimate of the improvement from considering longer routes may be obtained.

XFG also allows a choice between *cubic spline interpolation* and *linear interpolation* of the revenue function. Recall that a convex hull is fitted to each revenue function and the hull points are stored in a lookup table. In general linear interpolation is sufficient, but cubic spline interpolation is usually used to yield a continuous hull and it incurs only a small additional processing cost.

For some networks, XFG terminates with several *trivial routes* containing only a very few circuits (around one or two units of capacity). In practice, the setup costs for such routes outweigh their revenue advantage. XFG therefore has a parameter allowing all such routes to be discarded. As expected, experiments have confirmed that these routes contribute very little to the total rate of earning revenue, except in the very smallest of networks.

Finally, there is a parameter which enables the *randomization of tie breaks* within Dijkstra's algorithm. Recall that Dijkstra's algorithm is used to find the least cost routes between all O-D pairs. In its standard form, the algorithm selects the first route it finds which has the least cost. However, in general it is better to randomize the selection where several routes tie for the least cost – this was observed to result in better utilization of capacity.

4.5 Optimal Reconfiguration of Large Manhattan Grids

Consider the network presented in figure 4.3 consisting of 49 nodes connected by 84 links. Each link carries traffic in both directions. The network carries 6 services. The bandwidth requirement of the 1st service is 1 unit (64 kbps) and the bandwidth requirements of services 2 through 6 are 3, 4, 6, 24 and 40 units respectively (the term unit and circuit are synonymous). The capacity of each link is 10 000 units. The traffic intensity offered to O-D pair (o, d)

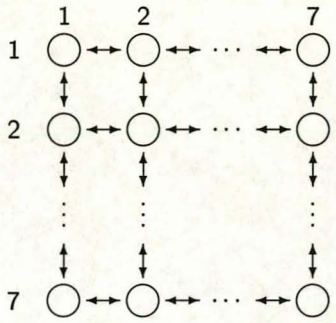

Figure 4.3: 49-node Manhattan grid

is $(\rho^1_{o,d}, \cdots, \rho^6_{o,d})$ where

$$\rho^s_{o,d} = \begin{cases} 1.0 & s = 3 \\ 0.5 & s = 1, 2, 4, 5, 6 \end{cases} \tag{4.7}$$

A call carried from o to d (or d to o) earns revenue at a rate which is independent of how the call is routed and is equal to the bandwidth of the call.

The network is initially configured with 10 000 units of capacity on each of the 84 physical links. At each step the XFG algorithm identifies the most profitable route r in the network: x units of capacity are cross connected along r and are added to the logical direct link connecting the origin and destination nodes of the route r. For this model, the reconfiguration unit $x = 20$.

The algorithm converges in 17 980 steps requiring 12 minutes of execution on a Pentium Pro 200 MHz processor. 4 520 routes are constructed and 148 902 units of capacity are allocated on these routes to form 1 092 VPs. The lengths of the routes are shown in table 4.1. The un-normalized length n_r of a route r connecting nodes o and d is equal to the number n_r of links in the route. The normalized length $\widehat{n}_r$ of the route r is given by $\widehat{n}_r = n_r - a_r$ where a_r is the number of links in the shortest route connecting nodes o and d. With reference to table 4.1, L_i $(\widehat{L}_i)$ is the number of routes of un-normalized (normalized) length i. The capacity distribution is also

route length	routes		capacity	
	L_i	$\widehat{L}_i$	C_i	$\widehat{C}_i$
0		3 856		139 800
1				
2	214	378	25 680	5 490
3	391		26 418	
4	621	221	25 376	2 816
5	729		21 846	
6	682	52	17 446	604
7	597		12 634	
8	471	13	8 316	192
9	364		5 594	
10	244		3 600	
11	132		1 572	
12	56		356	
13	17		58	
14	1		4	
15			0	
16	1		2	
total	4 520	4 520	148 902	148 902

Table 4.1: Distribution of: un-normalized L_i and normalized $\widehat{L}_i$ route lengths; capacity per un-normalized and normalized route length

shown in table 4.1 where C_i ($\widehat{C}_i$) is the capacity on routes of un-normalized (normalized) length i. The normalized distributions presented in table 4.1 reveals that 85% of the routes and 94% of the capacity in the optimal VPCN are on the shortest possible routes between their corresponding O-D pairs.

At each step the algorithm will release 2 units of capacity along one route if this leads to an improvement in the rate of earning revenue. The release mechanism allows the algorithm to extricate itself from non-optimal solutions. Once the first release occurs, the value of the reconfiguration unit x is decreased from 20 to 2. If all the capacity on a route is released, the route is said to be *extinguished*. Releases are made on 2 375 routes: 1 116 of these routes are extinguished. Table 4.2 reveals that, for this model, releases yield a small improvement in network revenue.

After reconfiguration the network is logically fully connected. The initial rate of earning revenue and the initial network blocking probability were 3 444 and 92.9% respectively. The optimal rate of earning revenue and the

optimal VPN	release	
	no	yes
routes	3 880	4 520
route capacity	146 720	148 902
routes released		2 375
routes extinguished		1 116
rate of earning revenue	45 478	45 684
blocking probability	5.7%	5.3%

Table 4.2: XFG reconfiguration

4220	800	140	140	440	3860
2620	340	140	140	340	2320
1480	260	140	140	280	1520
3680	580	140	140	700	3660
2460	280	140	140	260	2240
3120	340	140	140	340	3440
4700	860	140	140	1060	4800

4460	*2480*	*1480*	*3140*	*1900*	*2620*	*4020*
1080	*340*	*260*	*360*	*280*	*340*	*480*
140	*140*	*140*	*140*	*140*	*140*	*140*
140	*140*	*140*	*140*	*140*	*140*	*140*
800	*340*	*280*	*500*	*260*	*340*	*860*
4380	*3400*	*2500*	*3220*	*2640*	*3340*	*4540*

Figure 4.4: Capacities of direct routes after XFG reconfiguration (a) east–west (b) north–south

optimal network blocking probability are 45 684 and 5.3% respectively.

Figure 4.4 presents the capacities reserved for the routes connecting adjacent nodes. The capacities are rounded up to the nearest 20. The allocated capacities display an (approximate) symmetry about the vertical, horizontal and diagonal axes bisecting the grid.

At each step, the algorithm adjusts the capacity on a route. Each new route is assigned a unique label. The route selected at step 1 is assigned the label 1. The route selected at step n, if it is not among the m routes previously selected by the algorithm, is assigned the label $m + 1$.

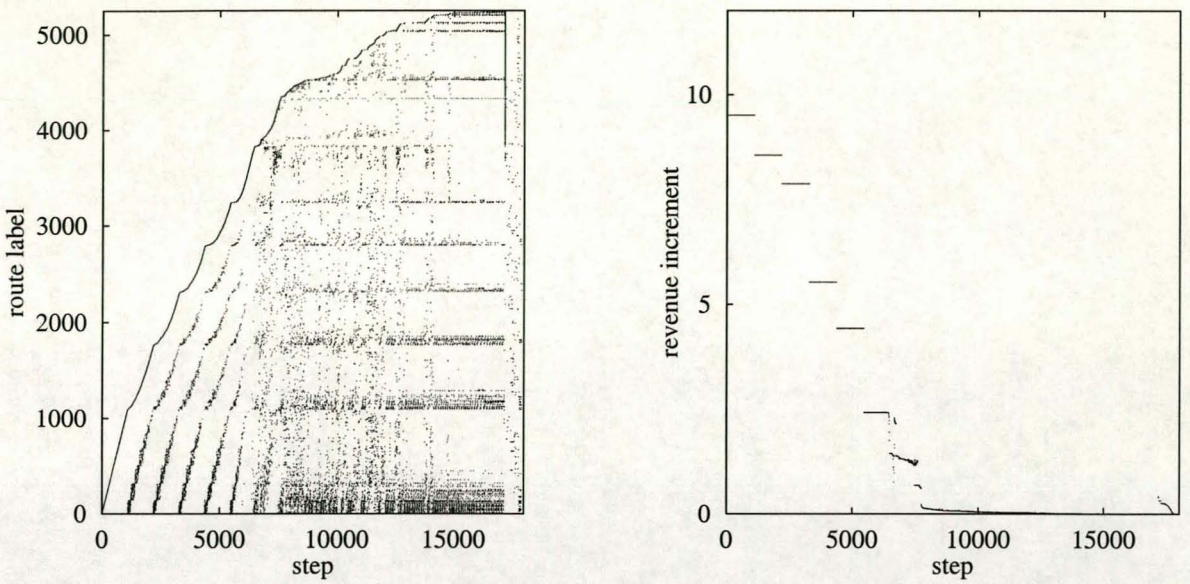

Figure 4.5: (a) Route selection (b) Incremental revenue earned per step

Figure 4.5a plots the label m of the route selected at step n. The symmetry of the network configuration and the symmetry of the offered traffics determine that routes are selected in cycles of 1 092 steps. During the first cycle the algorithm constructs a VP of capacity 20 for each of the 1 092 O-D pairs that are not directly connected. During the second cycle the 1 092 VPs receive 20 additional bandwidth units using capacity from new routes identified in cycle 2 and from routes used in cycle 1. The process is repeated: during the 6th cycle the VPs receive 20 additional units using capacity from new routes identified in cycle 6 and from routes discovered in cycles 1 through 5. Each VP now has capacity 120. The pattern of route re-use is clearly shown in figure 4.5a.

The pattern of route usage changes after the 6th cycle at step 6 552. The free capacity (capacity not assigned to VPs) on most physical links is now small enough to cause the link cost to rise substantially. The allocation unit is therefore reduced from 20 to 2 bandwidth units. The algorithm continues to find new routes and re-use previous routes. The routes now form distinct *working sets* which are illustrated by the horizontal bands in figure 4.5a. Each working set consists of routes which are used by the algorithm. The

memberships of the working sets change slowly as the algorithm executes. Immediately before termination, the algorithm *discards* (extinguishes) all routes that have only the minimal allocation of 2 units of capacity. The freed capacity is distributed among the remaining routes. Discards commence at step 19 550 whereupon the the working sets break down.

Figure 4.5b shows the change $\Delta F(\mathbf{x}_n)$ in the rate of earning revenue accomplished at each step of the algorithm,

$$\Delta F(\mathbf{x}_n) = F(\mathbf{x}_n) - F(\mathbf{x}_{n-1}). \tag{4.8}$$

The revenue increment is approximately constant within each cycle and decreases from cycle to cycle. The pattern is disrupted at step 6 654 where releasing commences, thereafter the revenue increments slowly approach zero. Immediately prior to termination, the reallocation of discarded capacity is seen to earn a small revenue increment.

Figure 4.6 displays the normalized lengths of the routes selected as the algorithm executes. Routes of (normalized) length 0 are selected throughout the execution of the algorithm. From step 6 654 onwards, when the cost of the physical links has increased sufficiently, the allocation unit is reduced from 20 to 2 units of bandwidth and the algorithm selects longer routes.

Figure 4.8a displays the capacity allocated on the direct routes (which have un-normalized length $n = 1$) and on the cross connected routes (which have length $n > 1$). Figure 4.4 shows that those direct routes located on the boundary of the grid are relatively underutilized. Figure 4.8b displays the VP capacity as a function of the normalized route length. 98% of the VP capacity is composed from the shortest possible routes connecting the O-D pairs.

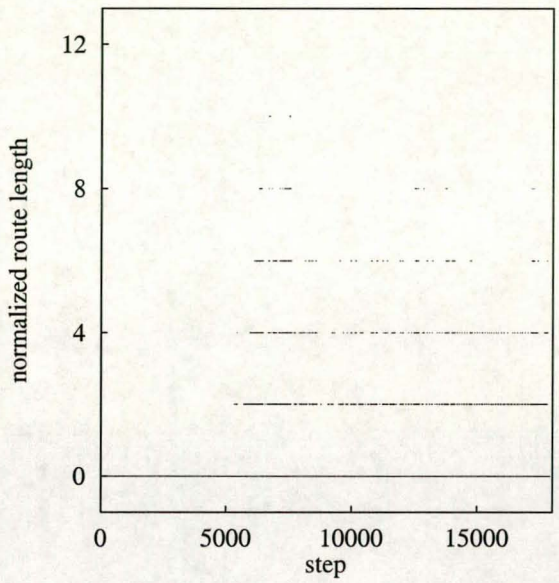

Figure 4.6: Normalized lengths of cross connected routes

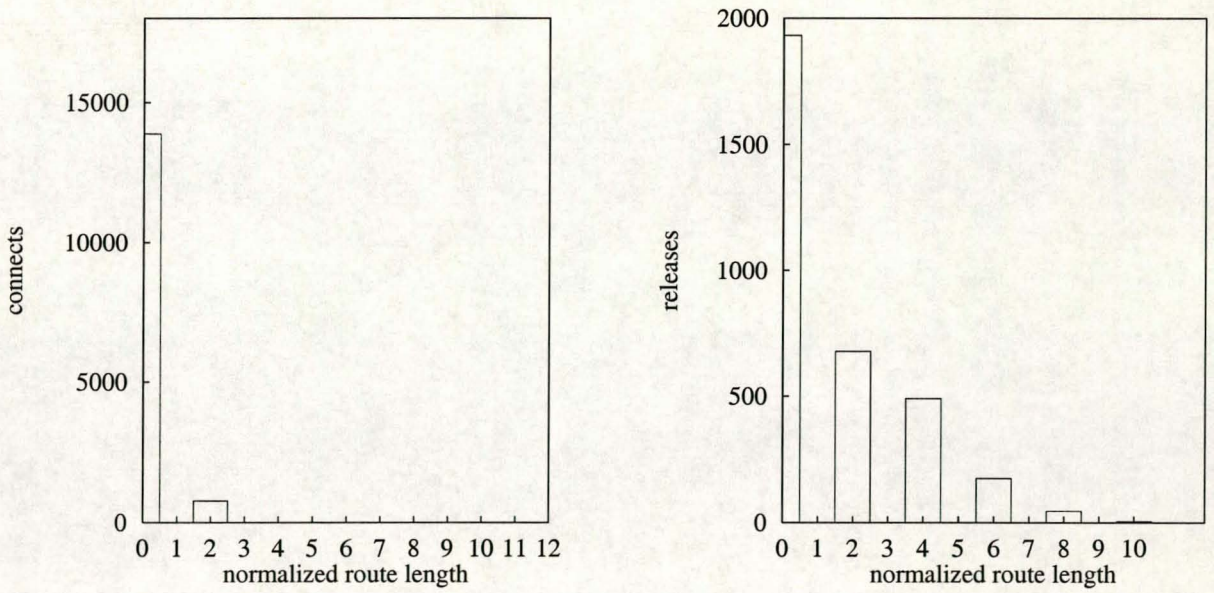

Figure 4.7: Lengths of (a) cross connected routes (b) released cross connections

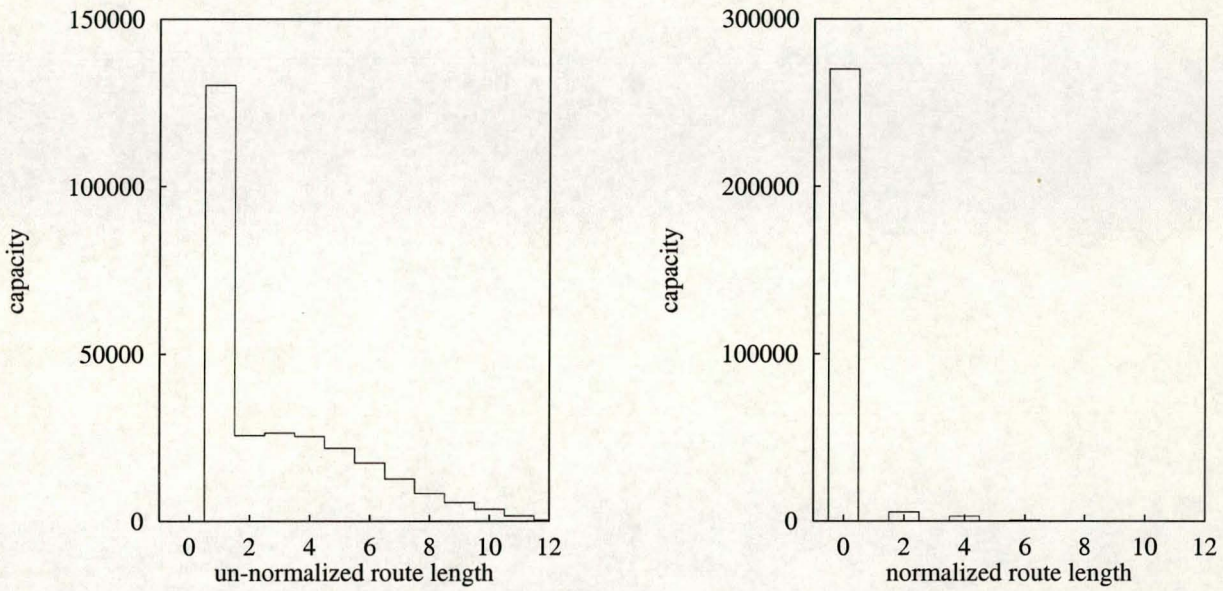

Figure 4.8: Route capacities by route length (a) un-normalized and (b) normalized

Chapter 5

XFG: Experimental Results

The XFG algorithm and a Markov chain simulator were used to compute the GOS attained by combinations of VPCN redesign, dynamic alternative routing (DAR) [14] and call queueing (see below). Since we did not have access to real traffic data, we first present a method for generating network data (a more detailed description appears in appendix A).

5.1 Network Data

A computer program [1, 19] was used to generate a series of networks and traffics. Each network consists of $N = 20$ nodes and $L = 48$ links carrying $S = 6$ service classes. Each link carries traffic in both directions. The links are place so that, among other features, all nodes are reachable (see appendix A). Calls of class 1 have an equivalent bandwidth requirement of $d^1 = 1$ unit of capacity and the equivalent bandwidth requirements of services 2 through 6 are 3, 4, 6, 24 and 40 units respectively.

Node pairs are connected to form physical links, starting with the two nodes closest to each other and moving up, until each node is reachable from every other node. For each O-D pair (o, d), an amount of capacity uniformly distributed in the range $[40, 500]$ is assigned to the physical link joining them or, if they are not physically connected, to the each of the links in the shortest route joining them. The traffic intensity $\rho_{o,d}$ offered to each O-D

pair (o, d) is set so that the loss probability averaged over all service classes on the VPC connecting nodes o and d is $y\%$. In our experiments we chose $y = 1$ and $y = 5$. The most highly connected $N/4$ nodes are chosen to form a set of busy nodes; the remaining nodes form a set of non-busy nodes. The busy set might represent nodes in an urban area and the non-busy set might define nodes in a suburban area. For each O-D pair (o, d) the offered traffic $\rho_{o,d}$ is multiplied by a factor s_{bb}, s_{bn}, s_{nn} according to whether both, one or none of the nodes are in the busy set.

The multiservice traffics are formed as follows. The class s traffic intensity $\rho^s_{o,d}$ is given by multiplying $\rho_{o,d}$ by a factor b^s. We chose $b^3 = 1$ and $b^s = 0.5$ for $s = 1$, 2, 4, 5 and 6. The class dependent arrival rates and holding times are set so that higher bandwidth calls arrive infrequently but have longer holding times. We set the arrival rate $\lambda^s_{o,d} = \rho^s_{o,d}/d^s$ and the holding time $1/\mu^s_{o,d} = d^s$ where d^s is the effective bandwidth requirement of a call of class s. Calls of class 1 thus have a mean holding time of 1 time unit, and calls of class 6 have a mean holding time of 40 time units.

This procedure is used to define $J = 8$ networks each with 20 nodes and 48 links. Each network $j = 1, \ldots, J$ has a different physical connectivity and different link sizes and has a different traffic intensity matrix ρ^j where the (o, d)th element of ρ^j is $\rho_{o,d}$ scaled according to the busy set membership of the nodes.

Finally, for each network $j = 1, \ldots, J$ the program was used to generate $K = 8$ traffic matrices $\rho^{j,k}$ where $k = 1, \ldots, K$. Each $\rho^{j,k}$ is a perturbation of ρ^j and represents the traffic conditions for the jth network during the kth period of the day. For the first four networks $j = 1, \ldots, 4$ the perturbations were small – the total traffic offered to the network changed by about 2% from one busy period to the next – and for the other four networks $j = 4, \ldots, 8$ the perturbations were moderate – the successive busy period traffics differ by about 10%.

route length	un-normalized			normalized	
	all	active	circuits	R^m	x^m
$m = 0$				152	51 894
1				41	7 434
2	410	67	28 599	14	782
3	961	71	21 217	2	505
4	3 793	41	8 194	1	2
5	13 026	24	2 255		
6	38 330	7	352		
links	48	48	62 364	48	62 364
total	56 568	258	122 981	258	122 981

Table 5.1: Distributions of route lengths and capacities

5.2 Solution Characteristics

A network model described in section 5.1 was capacitated to a GOS $y = 5\%$ prior to the traffics being scaled according to the busy set membership of the nodes. The scaling factors used were $s_{bb} = 1.2, s_{bn} = 1.0, s_{nn} = 0.9$. This resulted in a physical network configured with 229 376 units of transmission capacity distributed among the 48 physical links.

The XFG algorithm was applied to design an optimal VPCN configuration for this network. The algorithm converged in $n = 91\,268$ steps using 90 seconds of CPU time on a Pentium II 400 MHz processor. The transmission capacity was configured into 122 981 logical end-to-end circuits which were allocated on the 48 physical links and 210 routes. The reconfiguration yields a fully meshed optimal VPCN where the 190 VPCs carry the offered traffic (after busy set scaling) to a GOS of 5.14%.

The lengths of the routes and the capacity assignments are shown in table 5.1. Here we distinguish between *normalized* and *un-normalized* route lengths. The un-normalized length of a route is equal to the number of links in the route. The normalized length of a route connecting nodes o and d is given by its un-normalized length less the number of links in the shortest route connecting nodes o and d. The shortest routes thus have normalized lengths of zero.

As stated in section 3.3 of chapter 3, the XFG algorithm considers *all* possible routes and identifies and works with *active* routes, namely routes for which $x_r > 0$. With reference to table 5.1, the column labeled *all* shows the total number of routes in the network that are m-links long; the column labeled *active* shows the number of active routes that are m-links long; the column labeled *circuits* shows the number of end-to-end circuits allocated on m-link routes; R^m is the number of active routes of normalized length m; and x^m is the number of end-to-end circuits allocated on routes of normalized length m. The key observations are that only 210 of the 56 520 possible routes are active and that 99% of the logical circuits in the optimal VPCN are configured on the physical links and on routes with normalized lengths of 0 and 1.

The physical network presented above is sparse so that many of the shortest routes are more than 2-links long. If the route set is truncated to include only 2-, 3- and 4-link routes, the VPCN design problem will have some 5 000 decision variables. Standard NLP solvers may find it difficult to solve a problem of this size. However, the XFG algorithm can easily be adapted to work with the combinatorial optimization method of simulated annealing, see [27] and the references therein. When applied to the network under consideration, the SA method computes a revenue rate that is within 0.5% of the (near) optimal revenue computed by XFG. Note that the XFG solution requires some 90 seconds of CPU time and the SA solution requires about 30 minutes.

5.3 Bandwidth Redistribution

In this section we investigate the relationship between changes in traffic demand and changes in bandwidth allocation. Let $\rho^{j,k}$ denote the traffic intensity offered to the jth network during the kth time period

$$\rho^{j,k} = \sum_{o,d} \rho_{o,d}^{j,k} \tag{5.1}$$

k	$\rho^{j,k}$	$\sigma(\rho^{j,k})$	$C^{j,k}$	$\sigma(C^{j,k})$	$F(\mathbf{y}_{\text{opt}}^{j,k})$	$\epsilon^{j,k}$ %	GOS %				
							fixed		predetermined		
							direct	DAR	direct	DAR	queue
0	86303		121860		83237		1.49	0.54	1.49	0.54	0.56
1	91826	6803	118554	7325	86761	0.00	10.09	3.87	2.40	1.32	1.24
2	84562	8740	119656	8795	81708	0.00	8.84	3.02	1.49	0.48	0.61
3	88457	8730	120327	8495	83048	0.00	9.40	3.53	2.61	1.55	1.94
4	77342	9088	118420	8872	76212	0.00	6.55	1.43	0.60	0.11	0.12
5	91333	8590	121751	8530	85481	0.01	10.47	4.57	2.77	1.74	1.78
6	93120	9331	127353	9473	86582	0.00	9.27	4.34	3.16	1.85	2.13
7	91597	8930	122039	9545	85865	0.06	8.41	4.08	2.83	1.55	1.80
8	83865	7018	119275	7261	80727	0.00	7.01	2.13	1.61	0.55	0.64
avg	87763	8404	120922	8537	83343	0.01	8.76	3.37	2.18	1.14	1.28

Table 5.2: Performance averaged over all service classes for various call routing strategies and fixed and predetermined VPCN management

where $\rho_{o,d}^{j,k}$ is the traffic intensity offered to the O-D pair (o,d) during the kth time period. Let

$$\sigma^2(\rho^{j,k}) = \sum_{o,d}(\rho_{o,d}^{j,k} - \rho_{o,d}^{j,k-1})^2 \tag{5.2}$$

denote a measure of the difference between the traffic intensity offered to network j during traffic periods k and $k-1$. Similarly, let

$$C^{j,k} = \sum_{o,d} C_{o,d}^{j,k} \tag{5.3}$$

denote the capacity of all the VPCs in network j during traffic period k, and let

$$\sigma^2(C^{j,k}) = \sum_{o,d}(C_{o,d}^{j,k} - C_{o,d}^{j,k-1})^2 \tag{5.4}$$

denote a measure of the difference between the capacity of network j during traffic periods k and $k-1$. We expect that variable demands will be reflected in varying supplies.

Table 5.2 presents several results from the reconfiguration of a multiservice network that is offered a sequence of 8 busy hour traffics. The physical network and the offered traffics are generated according to the procedure described in section 5.1. The network is configured so that 1% of the calls are lost prior to the traffics being scaled according to the busy set membership of the O-D pairs. After scaling, 1.49% of the calls are lost. The ratios $\sigma(\rho^{j,k})/\rho^{j,k}$ where $k = 1,\ldots,8$ show that the total traffic intensity changes by about 10% from one busy period to the next. The VPCN is

reconfigured to earn the optimal revenue from each busy period. The ratio $\sigma(C^{j,k})/C^{j,k}$ shows that about 10% of the network's end-to-end capacity is reallocated from one busy period to the next in response to the changing traffics. Note that $\sigma(\rho^{j,k}) \sim \sigma(C^{j,k})$ indicating that VPC capacities are adjusted in proportion to the changes in the traffic intensity.

5.3.1 Initial Conditions

The XFG algorithm yields a globally optimal configuration where the route capacities x_r are real valued. However, our implementation of the XFG algorithm computes integer valued route capacities. We therefore first confirm that the integer valued implementation of the XFG algorithm computes a (near) optimal configuration which is independent of the route configuration chosen to initialize the algorithm. For each network j and each traffic condition k we compute

- the optimal configuration $\mathbf{x}_{\text{opt}}^{j,k}$ when the integer valued XFG calculation begins from an initially empty configuration $\mathbf{x}_0^{j,k} = 0$, and
- the optimal configuration $\mathbf{y}_{\text{opt}}^{j,k}$ when the integer valued XFG calculation begins from a configuration $\mathbf{x}_0^{j,k}$ which is initialized to the previous optimal configuration $\mathbf{x}_{\text{opt}}^{j,k-1}$.

Let

$$\epsilon^{j,k} = \frac{F(\mathbf{x}_{\text{opt}}^{j,k}) - F(\mathbf{y}_{\text{opt}}^{j,k})}{F(\mathbf{x}_{\text{opt}}^{j,k})} \tag{5.5}$$

denote the relative error when the calculation of the kth design for network j is initialized from the previous optimal $(k-1)$st design. We expect that the choice of the initial configuration will not affect the value of the optimal configuration, and that the kth design will be computed faster if its computation is initialized from the previous optimal $(k-1)$st design.

Table 5.2 shows that for the models under consideration $\epsilon^{j,k} \sim 0$ which confirms that the calculation of the optimal VPCN is almost independent

of the initialization of the integer valued implementation of the XFG algorithm. Some 90 seconds of CPU time are required to compute an optimal configuration when the XFG calculation is started from an empty route set, and about 20 seconds are required if the kth design is initialized from the previous optimal $(k-1)$st design. The calculations were performed on a 400-MHz Pentium II processor.

5.4 VPCN and VC Management

For each network j and for each traffic period k we compare the network revenue rates $F(\mathbf{x}_{\text{opt}}^{j,k})$ and $F(\mathbf{x}_{\text{opt}}^{j})$ earned when the traffic demand $\rho^{j,k}$ is offered to

- a configuration $\mathbf{x}_{\text{opt}}^{j}$ so that the k^{th} VPCN is designed according to the average traffic demand ρ^{j}, and
- the optimal configuration $\mathbf{x}_{\text{opt}}^{j,k}$: in this case the kth VPCN is tailor made for the traffic demand $\rho^{j,k}$.

These two cases correspond to different VPCN management strategies: the *fixed* and the *predetermined* approaches respectively. We expect that predetermined VPCN management will improve the network revenue.

We next evaluate several routing schemes for both strategies, namely

- *Direct routing* where calls use the VPCs only.
- *Dynamic alternative routing* (DAR) [14] where a blocked call will attempt to be connected via an alternative route consisting of two VPCs from the origin via an arbitrary node to the destination node. Alternative routing is combined with trunk reservation such that VPCs do not accept alternatively routed calls unless both VPCs in the alternative route have a certain minimum free capacity (the reservation parameter). The reservation parameter is set to 4 for all services.

- *Call queuing* where the signalling units for a blocked call are queued in a small buffer. The call will be connected after some delay. The call is lost if the buffer is full. Each VPC has its own buffer which is shared among all service classes using the VPC. The buffer can store 10 calls. When a call completes the corresponding queue is served: queued calls are connected in random order until the freed capacity is exhausted.

We expect that DAR and call queuing will improve performance by handling the short lived random mismatches between traffic and capacity.

Table 5.2 presents the GOS in percent (averaged over all services) that is achieved for the various combinations of VPCN management and call routing. We observe that a fixed VPCN results in an unacceptable GOS: the GOS averaged over the 8 busy periods is 8.8%. Applying DAR to the fixed VPCN offers a significant improvement: the average GOS is now 3.4%. Redesigning the VPCN to meet the changing traffic conditions results in an average GOS of 2.2%. Adding DAR to redesigned VPCNs allows the network to manage random traffic changes as well: this additional flexibility improves the GOS to 1.1%. Adding call queueing to redesigned VPCNs results in a GOS of 1.3%.

It would appear from table 5.2 that, for the network under consideration, redesign offers a modest improvement over a fixed design combined with DAR insofar as the *average* GOS is concerned. However, we now show that VPCN redesign, particularly when combined with DAR or call queueing, offers a significant improvement to the GOS afforded to wideband calls.

Table 5.3 examines how the various combinations of VPCN management and call routing are able to deal with a multiservice workload. Table 5.3a shows that when the network is configured to offer a GOS of 1% prior to busy set scaling, DAR significantly reduces the blocking probability for the narrowband calls and offers a substantial improvement for the wideband calls: 74% of the highest bandwidth calls that attempt alternative routing are connected.

(a) network configured to 1% GOS

	blocking probabilities %							
service	all	1	2	3	4	5	6	revenue
bandwidth	–	1	3	4	6	24	40	
direct	1.04	0.08	0.24	0.31	0.47	2.14	3.76	83 715
DAR	0.36	0.00	0.01	0.01	0.03	0.65	1.82	84 963
success %	91	97	98	97	96	83	74	
queue	0.37	0.24	0.35	0.38	0.39	0.44	0.44	85 526
delay	0.18	0.14	0.28	0.44	0.78	2.64	7.97	

(b) network configured to 5% GOS

	blocking probabilities %							
service	all	1	2	3	4	5	6	revenue
bandwidth	–	1	3	4	6	24	40	
direct	4.74	0.93	1.66	2.00	2.74	9.26	14.56	98 656
DAR	4.03	0.14	0.33	0.42	0.74	8.26	17.87	97 598
success %	80	90	89	89	87	57	40	
queue	4.55	2.80	4.25	4.55	4.78	5.47	5.44	104 200
delay	0.14	0.10	0.21	0.33	0.60	3.17	13.37	

Table 5.3: Performance per service class for various call routing strategies and predetermined VPCN management

In addition, table 5.3a shows that call queueing has a fairness property: the low bandwidth calls are penalized and the high bandwidth calls are afforded a much improved GOS, but the waiting delay for high bandwidth calls is substantial (the waiting delay is expressed in units of holding time, where a class 1 call has unit average holding time). However, a customer may wish to wait in order to be connected, and the network operator may want to offer this possibility in order to earn revenue, in particular since wideband calls generate a large revenue. The call completion rate for narrowband calls is much higher than the call completion rate for wideband calls, and thus wideband calls have a large waiting time. A more advanced queueing mechanism which postpones the service of narrowband calls in order to afford better service to wideband calls, and which allows calls to depart without being served would reduce the waiting time for wideband calls.

These characteristics are emphasized when the network is configured to a GOS of 5% (pre-scaling) and is thus heavily loaded. Table 5.3b shows that DAR again significantly reduces the blocking probability for the low band-

width calls, but the performance of the high bandwidth calls is degraded: only 40% of the highest bandwidth calls that attempt alternative routing are connected, and the network revenue is degraded. Call queueing penalizes the low bandwidth calls, but the high bandwidth calls are now afforded a much improved GOS, leading to a significantly improved revenue. The waiting delay for the high bandwidth calls is again substantial.

Finally, table 5.4 shows how the various combinations of VPCN management and call routing are able to deal with the sequence of busy hour multiservice traffics. Note that tables 5.3 and 5.4 refer to two different networks j generated by the procedure described in section 5.1. Table 5.4a presents several results for a network configured so that 1% of the calls are lost prior to the traffics being scaled according to the busy set membership of the O-D pairs. After scaling 1.49% of the calls are lost. The table shows that the GOS is unacceptable when the VPCN is not reconfigured. On average some 18.8% of revenue offered to the network is lost. The wideband calls (classes 5 and 6) receive poor service.

Table 5.4b shows that DAR significantly reduces the blocking probability for the narrowband calls (classes 1 through 4) and offers a substantial improvement for the wideband calls. Some 9.4% of the offered revenue is now lost. Table 5.4c shows that the redesign of the VPCN according to the busy hour traffic conditions offers a good GOS to the narrowband calls and a better service to the wideband calls. Some 5.0% of the offered revenue is lost.

Table 5.4d shows that, as expected, the addition of DAR to the redesigned VPCNs significantly reduces the blocking probability for all calls. Some 3.3% of the offered revenue is now lost. Table 5.4e shows that the addition of call queueing to the redesigned VPCNs tends to equalize the GOS over all service classes. Some 1.4% of the offered revenue is lost.

The key observations are that the combination of VPCN redesign with DAR or call queueing allows the network to meet its GOS target of 1.5% (after busy set scaling) during most of the traffic periods, and that the combination of VPCN redesign and call queueing achieves the desired GOS for the wideband calls, at the expense of a connection delay.

(a) fixed, direct

	GOS % (blocking/revenue loss)						
k	average	class 1	class 2	class 3	class 4	class 5	class 6
0	1.49/ 3.55	0.13	0.36	0.48	0.72	3.00	5.26
1	10.09/21.39	1.25	3.53	4.62	6.68	20.76	29.17
2	8.84/18.39	1.16	3.26	4.24	6.08	17.98	24.89
3	9.40/19.90	1.19	3.31	4.31	6.23	19.28	27.15
4	6.55/14.42	0.76	2.07	2.71	3.96	13.49	20.16
5	10.47/22.59	1.24	3.49	4.58	6.66	21.66	31.12
6	9.27/19.84	1.14	3.18	4.16	6.01	18.97	27.30
7	8.41/18.24	0.99	2.77	3.63	5.28	17.28	25.28
8	7.01/15.48	0.77	2.19	2.89	4.22	14.48	21.66

(b) fixed, DAR

0	0.54/ 1.58	0.01	0.03	0.03	0.06	0.98	2.64
1	3.87/10.73	0.09	0.30	0.40	0.75	8.34	16.78
2	3.02/ 8.41	0.07	0.24	0.31	0.57	6.39	13.24
3	3.53/ 9.81	0.08	0.27	0.36	0.62	7.66	15.33
4	1.43/ 4.07	0.03	0.09	0.12	0.21	2.94	6.52
5	4.57/12.74	0.11	0.33	0.46	0.85	9.74	20.02
6	4.34/12.07	0.10	0.34	0.44	0.84	9.26	18.94
7	4.08/11.43	0.10	0.30	0.41	0.72	8.46	18.15
8	2.13/ 6.05	0.04	0.12	0.18	0.34	4.33	9.71

(c) predetermined, direct

0	1.49/ 3.55	0.13	0.36	0.48	0.72	3.00	5.26
1	2.40/ 5.52	0.28	0.67	0.87	1.28	4.76	8.05
2	1.49/ 3.37	0.23	0.45	0.57	0.82	2.89	4.91
3	2.61/ 6.12	0.24	0.68	0.90	1.35	5.31	8.91
4	0.60/ 1.46	0.04	0.13	0.17	0.27	1.18	2.21
5	2.77/ 6.42	0.28	0.75	0.99	1.47	5.57	9.34
6	3.16/ 7.02	0.53	1.02	1.26	1.77	6.14	10.10
7	2.83/ 6.20	0.55	0.97	1.17	1.61	5.40	8.93
8	1.61/ 3.74	0.18	0.44	0.56	0.82	3.19	5.49

(d) predetermined, DAR

0	0.54/ 1.58	0.01	0.03	0.03	0.06	0.98	2.64
1	1.32/ 3.80	0.02	0.08	0.10	0.19	2.59	6.18
2	0.48/ 1.39	0.01	0.03	0.04	0.06	0.87	2.30
3	1.55/ 4.45	0.03	0.08	0.13	0.23	2.99	7.26
4	0.11/ 0.32	0.00	0.00	0.00	0.01	0.17	0.56
5	1.74/ 5.00	0.03	0.10	0.14	0.25	3.40	8.13
6	1.85/ 5.23	0.06	0.13	0.17	0.30	3.62	8.46
7	1.55/ 4.37	0.06	0.13	0.17	0.27	3.00	7.08
8	0.55/ 1.59	0.01	0.03	0.04	0.07	1.02	2.63

(e) predetermined, call queueing

0	0.56/ 0.64	0.35	0.54	0.56	0.56	0.65	0.68
1	1.24/ 1.36	0.80	1.20	1.29	1.33	1.44	1.37
2	0.61/ 0.67	0.41	0.59	0.62	0.63	0.69	0.68
3	1.94/ 2.22	1.17	1.85	1.98	2.06	2.22	2.34
4	0.12/ 0.13	0.08	0.13	0.13	0.14	0.15	0.12
5	1.78/ 2.00	1.09	1.68	1.80	1.90	2.13	2.04
6	2.13/ 2.36	1.40	2.09	2.19	2.20	2.50	2.38
7	1.80/ 2.01	1.22	1.70	1.82	1.87	2.08	2.08
8	0.64/ 0.70	0.43	0.64	0.66	0.68	0.73	0.71

Table 5.4: Performance per service class for various call routing strategies and fixed and predetermined VPCN management

Chapter 6

Conclusion

This thesis is concerned with dynamic VPCN redesign as a resource management control in ATM networks where transmission capacity is reserved on the communication links in order to form dedicated logical paths for each origin-destination flow. We formulate the VPCN design problem in terms of a constrained nonlinear programming problem and propose a novel, efficient algorithm to solve it. Applying the results to a set of test networks, we show that dynamic VPCN reconfiguration can be advantageous where dealing with slow traffic variations in large networks. We show that the use of dynamic VPCN to deal with slow traffic variations, when combined with DAR or call queueing to deal with the short lived random mismatches between offered traffic and capacity, can achieve the required GOS. We note finally that for the networks being evaluated the combination of dynamic VPCN redesign and call queueing can achieve the desired GOS for the wideband calls, at the expense of a connection delay.

Appendix A

NetGen: Generating Random Networks and Traffic Data

This appendix describes a program we have developed called NetGen which generates (at least partially) realistic network configurations and corresponding multiservice call traffic data. The main use of the NetGen program is to investigate the efficiency of dynamic network reconfiguration [2] where a virtual path connection network (VPCN) is layered on top of a physical transmission network. The VPCN is configured to optimize the network revenue. The NetGen program is based on, and extends, a network generating algorithm by Arvidsson [1].

A.1 Introduction

NetGen is designed to synthesize multiservice networks which carry several classes of traffic with varying bandwidth requirements. NetGen assumes that the bandwidth requirements of the call classes are specified in terms of their equivalent bandwidths which are expressed in integer units of capacity analogous to circuits.

In brief, the algorithm randomly places N nodes on a canvas of dimension

$X \times Y$. Connections are made to ensure that all nodes are reachable and that the mesh factor is met (see below). Capacity sampled from a uniform distribution is allocated to the connections, and arrival intensities are chosen to meet a given blocking probability. An imaginary circle is drawn around the $^1/_4$ of the nodes that are closest together, defining a busy area. Traffics on O-D pairs with both nodes in the busy area are increased and those with neither node in the busy area decreased. Finally all traffics are perturbed.

The networks NetGen generates have several realistic features. Nodes that are closer together are more highly interconnected – this is typical, for example, of nodes in a densely populated area. The selection of a busy area and the resulting traffic scaling produce network data corresponding to typical busy hour traffic patterns where calls are concentrated in an urban area. The generator also distinguishes between the base network (the network topography, including link capacities) and a specific network instance (with traffics perturbed). This allows the generation of a sequence of network descriptions, all with the same layout, but with different traffic call patterns. Such a sequence represents changing traffic, at a time scale where such changes are slow.

From the perspective of the NetGen program, the following parameters are used to describe a network

- The set of N nodes and their geographical coordinates. The coordinates may be used by a visualization system to give a geographical view of the network.
- A set of L_M origin–destination (O-D) pairs (i, j) and their offered traffic intensities ρ_{ij}.
- A set of L physical links and their capacities where C_{ij} denotes the number of circuits on link (i, j). Let $L_M = N(N-1)/2$ denote the maximum number of possible links. The ratio $\gamma = L/L_M$ is called the mesh factor.
- The number of call classes K, their bandwidth requirements b_k and

traffic intensity factors β_k. Class k calls attempting to connect on (i, j) each require b_k resource units and have traffic intensity $\beta_k \rho_{ij}$.

A.2 Algorithm Description

The NetGen program takes as input

r_1, r_2 two integers used to seed the random number generator at different stages of the algorithm

K the number of service classes

$b_1, \ldots, b_K$ the bandwidth requirements of each class

$\beta_1, \ldots, \beta_K$ the class-dependent traffic intensity factors

N the number of nodes in the network

X, Y the width (X) and height (Y) of the area in which nodes are placed, in arbitrary units

γ the mesh factor

B the initial blocking probability for all O-D pairs

s_{xy} scaling factors for traffics between busy–busy (s_{bb}), busy–normal (s_{bn}) and normal–normal nodes (s_{nn})

p a perturbation factor.

The NetGen program proceeds as follows

1. Seed the random number generator with r_1.
2. Network Topology.

   Place the N nodes over a surface of size $X \times Y$ at random coordinates (x_n, y_n).

Compute the Euclidean distances between all pairs of nodes. Add one link at a time by joining the two not yet interconnected nodes i, j that are closest to each other, until each node is reachable from any other node.

Add links to provide protection against a node and its associated links being down by removing one node at a time and checking for full connectivity among the remaining nodes. If islands occur, then for each pair of islands, identify their members and connect the two nodes that are closest to each other, yet members of different islands.

Add one link at a time by joining the two not yet directly connected nodes i, j that are the closest to each other, until γL_M links have been added, where γ is the mesh factor.

3. Routing Pattern.

   Compute the shortest routes between all O-D pairs using Dijkstra's algorithm [7].

4. Transmission Capacities.

   Initially define the capacities of all links to be zero.

   For each O-D pair define a capacity, M_{ij}, uniformly distributed between b_K and 500. Add this to the capacities of all links belonging to the shortest route connecting the O-D pair.

5. Basic Traffics.

   For each O-D pair (i, j) assign an offered traffic intensity such that the expected loss will be exactly B given the transmission capacity associated with the O-D pair M_{ij}. To do this, Brent's method of root finding [27, Chapter 9] is applied to the multiservice version of Erlang's formula with C_{ij}, $b_1, \ldots, b_K$, ρ_{ij}, $\beta_1, \ldots, \beta_K$ as inputs.

6. Specific Traffics.

   Seed the random number generator with r_2.

   Use a clustering algorithm to select the most highly connected $N/4$ nodes. These nodes form the set of busy nodes. The other nodes form

the set of non-busy nodes.

For each O-D pair multiply the traffics by s_{bb}, s_{bn}, s_{nn} according to whether both nodes are busy (s_{bb}); one is busy, one is not (s_{bn}); or neither is busy (s_{nn}).

For each O-D pair perturb the traffics by multiplying by $P(p) \equiv U(1-p, 1+p)$. Where $U(\cdot,\cdot)$ is a function returning a random number, sampled from a uniform distribution, which lies between its arguments.

The outputs of the NetGen program are

(x_n, y_n) the (x, y) coordinates of the randomly placed nodes

C_{ij} the capacities of the physical links

ρ_{ij} the traffic intensities for all O-D pairs.

Bibliography

[1] Å. Arvidsson. High level B-ISDN/ATM traffic management in real time. In D. D. Kouvatsos, editor, *Performance Modelling and Evaluation of ATM Networks*, volume 1, pages 177–207. Chapman & Hall, London, 1995.

[2] Å. Arvidsson, S. A. Berezner, and A. E. Krzesinski. Dynamic reconfiguration in ATM networks. In *Proceedings of the Sixth IFIP Workshop on Performance Modelling and Evaluation of ATM Networks*, July 1998.

[3] T. Bauschert. Multihour design of multi-hop virtual path base wide-area ATM networks. In *Proceedings of ITC-15 the Fifteenth International Teletraffic Congress*, Washington, D.C., USA, June 1997.

[4] N. G. Bean. Effective bandwidths with different quality of service requirements. In *Proceedings of IBCN&S*, Denmark, April 1993. Paper 13.3.

[5] S. A. Berezner and A. E. Krzesinski. Call admission and routing in ATM networks based on virtual path separation. In P. J. Kühn and R. Ulrich, editors, *Proceedings of the Fourth International Conference on Broadband Communications IFIP TC6/WG6.2*, pages 461–472, Stuttgart, Germany, April 1998. Chapman & Hall.

[6] S. A. Berezner, A. E. Krzesinski, and P. G. Taylor. On the inverse of Erlang's function. *Journal of Applied Probability*, 35:246–252, 1998.

[7] D. Bertsekas and R. Gallager. *Data Networks*. Printice Hall, 1992.

[8] COST 224: Performance evaluation and design of multiservice networks. European cooperation in the field of scientific and technical research, October 1991.

[9] COST 242: Multi-rate models for dimensioning and performance evaluation of ATM networks. European cooperation in the field of scientific and technical research, 1994.

[10] S. Evans. A mathematical model and related problems of optimal management and design in a Broadband Integrated Services Network. *Australian Mathematical Society. Journal B*, 31:150–175, 1989.

[11] A. Gersht and A. Shulman. Optimal routing in circuit switched communication networks. *IEEE Transactions on Communications*, 37(11):1203–1211, November 1989.

[12] R. Geuérin, H. Ahmadi, and M. Naghshineh. Equivalent capacity and its application to bandwidth allocation in high-speed networks. *IEEE Journal on Selected Areas in Communications*, 9(7):968–981, 1991.

[13] R. J. Gibbens and P. J. Hunt. Effective bandwidths for the multi-type UAS channel. *Queueing Systems*, 9:17–28, 1991.

[14] R. J. Gibbens and F. P. Kelly. Dynamic routing in fully connected networks. *IMA Journal of Mathematical Control & Information*, 7:77–111, 1990.

[15] G. Gopal, C.-K. Kim, and A. Weinrib. Algorithms for reconfigurable networks. In *Proceedings of ITC-13 the Thirteenth International Teletraffic Congress*, pages 341–347, Copenhagen, Denmark, 1991.

[16] G. Gopal, C.-K. Kim, and A. Weinrib. Dynamic network configuration management. In *Proceedings of the IEEE International Conference on Communication*, Atlanta, Georgia, U.S.A., 1991. paper no. 302.2.

[17] M. Herzberg and S. Byes. Bandwidth management for reconfigurable multi-service networks. In *Proceedings of the Eighth Australian Teletraffic Seminar*, pages 280–290, Melbourne, Australia, 1993.

[18] J. Y. Hui. Switching and traffic theory for integrated broadband networks. *IEEE Journal on Selected Areas in Communications*, SAC-6(9):1598–1608, December 1988.

[19] A. M. Inggs and A. E. Krzesinski. An algorithm for generating random network data. In *Proceedings of SATNAC 98 the First South African Telecommunications, Networks and Applications Conference*, pages 669–672, Cape Town, South Africa, September 1998.

[20] J. S. Kaufman. Blocking in a shared resource environment. *IEEE Transactions on Communications*, COM-29(10):1474–1481, October 1981.

[21] F. P. Kelly. Fixed point models of loss networks. *Australian Mathematical Society. Journal B*, 31:204–218, 1989.

[22] F. P. Kelly. Effective bandwidths in multi-class queues. *Queueing Systems*, 9:5–16, September 1991.

[23] F. P. Kelly. Loss networks. *The Annals of Applied Probability*, 1(3):319–378, 1991.

[24] D. Mitra and J. A. Morrison. Erlang capacity and uniform approximations for shared unbuffered resources. *IEEE/ACM Transactions on Networks*, 2:558–570, 1994.

[25] D. Mitra, J. A. Morrison, and K. G. Ramakrishnan. ATM network design and optimization: A multirate loss network framework. In *Proceedings of IEEE INFOCOM '96*, volume 3, pages 994–1002, San Francisco, USA, March 1996.

[26] M. Pióro and P. Gajowniczek. Stochastic allocation of virtual paths to ATM links. In *Proceedings of the Second IFIP TC6 Workshop on Performance Modelling and Evaluation of ATM Networks*, pages 135–146, Bradford, U.K., 1994.

[27] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 1988.

[28] J. W. Roberts. Teletraffic models for the Telecom 1 integrated services network. In *Proceedings of ITC-10 the Tenth International Teletraffic Conference*, Montreal, 1983.

[29] M. Røhne, T. Jensen, I. Svinnset, and R. Venturin. Designing VP networks. In *Proceedings of the Fourteenth Nordic Teletraffic Seminar*, pages 17–30, Copenhagen, Denmark, 1998.

[30] K. W. Ross. *Multiservice Loss Models for Broadband Telecommunication Networks.* Telecommunication Networks & Computer Systems Series. Springer–Verlag, 1995.

[31] E. Wallmeier. A connection acceptance algorithm for ATM networks base on mean and peak bit rates. *International Journal on Digital and Analog Communication Systems*, 3:144–153, 1990.